What's New in CloudKit

There's a lot we want to share

Session 226

Paul Seligman CloudKit Engineer
Jacob Farkas CloudKit Engineer
Vanessa Hong CloudKit Engineer

Table of Contents

What are we talking about?

Table of Contents

What are we talking about?

CloudKit Overview

Table of Contents

What are we talking about?

CloudKit Overview

Telemetry

Table of Contents

What are we talking about?

CloudKit Overview

Telemetry

API Improvements

Table of Contents

What are we talking about?

CloudKit Overview

Telemetry

API Improvements

Sharing

CloudKit Overview

What is CloudKit?

CloudKit Overview

Data everywhere!

CloudKit Overview

Data everywhere!

iCloud Database

CloudKit Overview

Data everywhere!

iCloud Database

Extensive use inside Apple

CloudKit Overview

Data everywhere!

iCloud Database

Extensive use inside Apple

Ubiquitous

CloudKit Overview

Prior talks

Introducing CloudKit	WWDC 2014

CloudKit Overview

Prior talks

Introducing CloudKit	WWDC 2014
Advanced CloudKit	WWDC 2014
CloudKit JS and Web Services	WWDC 2015
What's New in CloudKit	WWDC 2015
CloudKit Tips and Tricks	WWDC 2015

CloudKit Overview

Prior talks

Introducing CloudKit	WWDC 2014
Advanced CloudKit	WWDC 2014
CloudKit JS and Web Services	WWDC 2015
What's New in CloudKit	WWDC 2015
CloudKit Tips and Tricks	WWDC 2015

https://developer.apple.com/cloudkit

CloudKit Overview

Core objects

Container

Database

Record

Record Zone

CloudKit Overview

Core objects

Container

Database

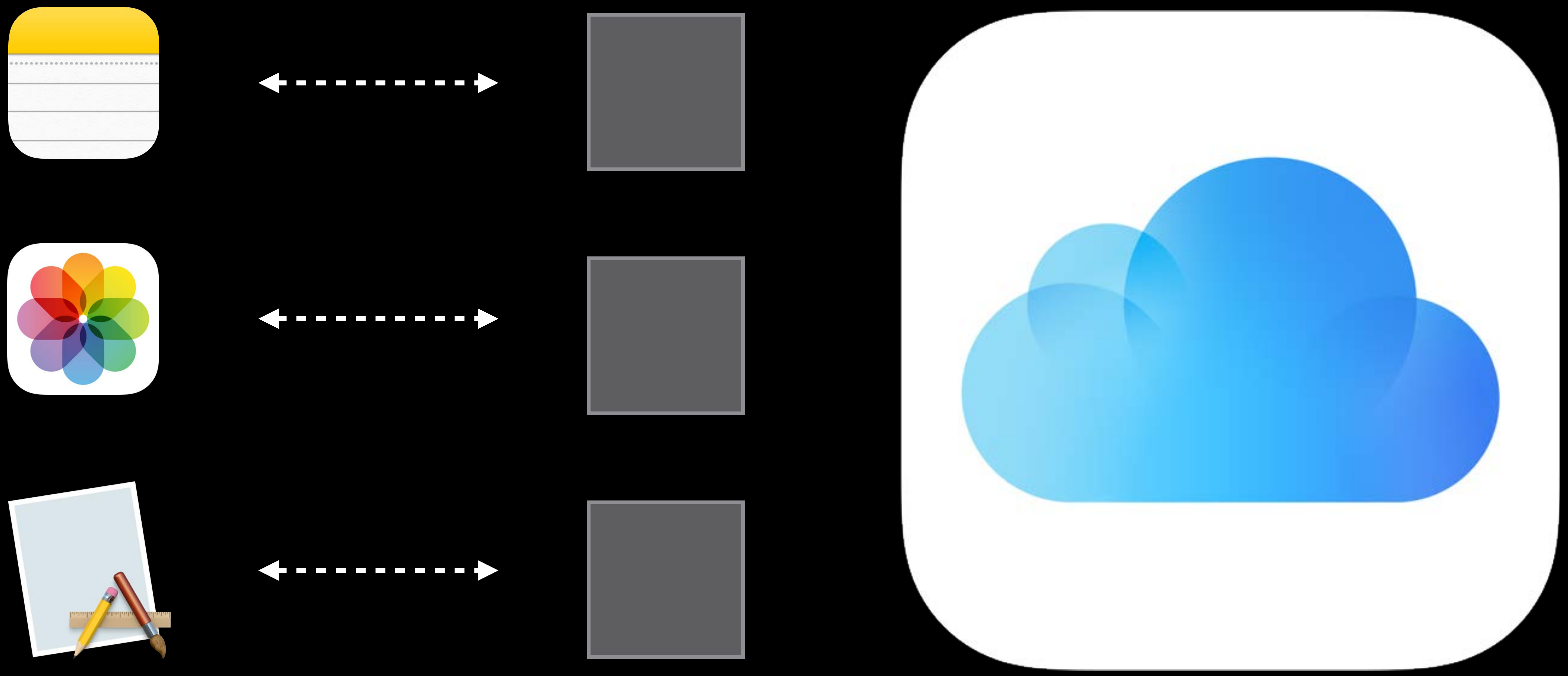
Record

Record Zone

CloudKit Overview

Core objects

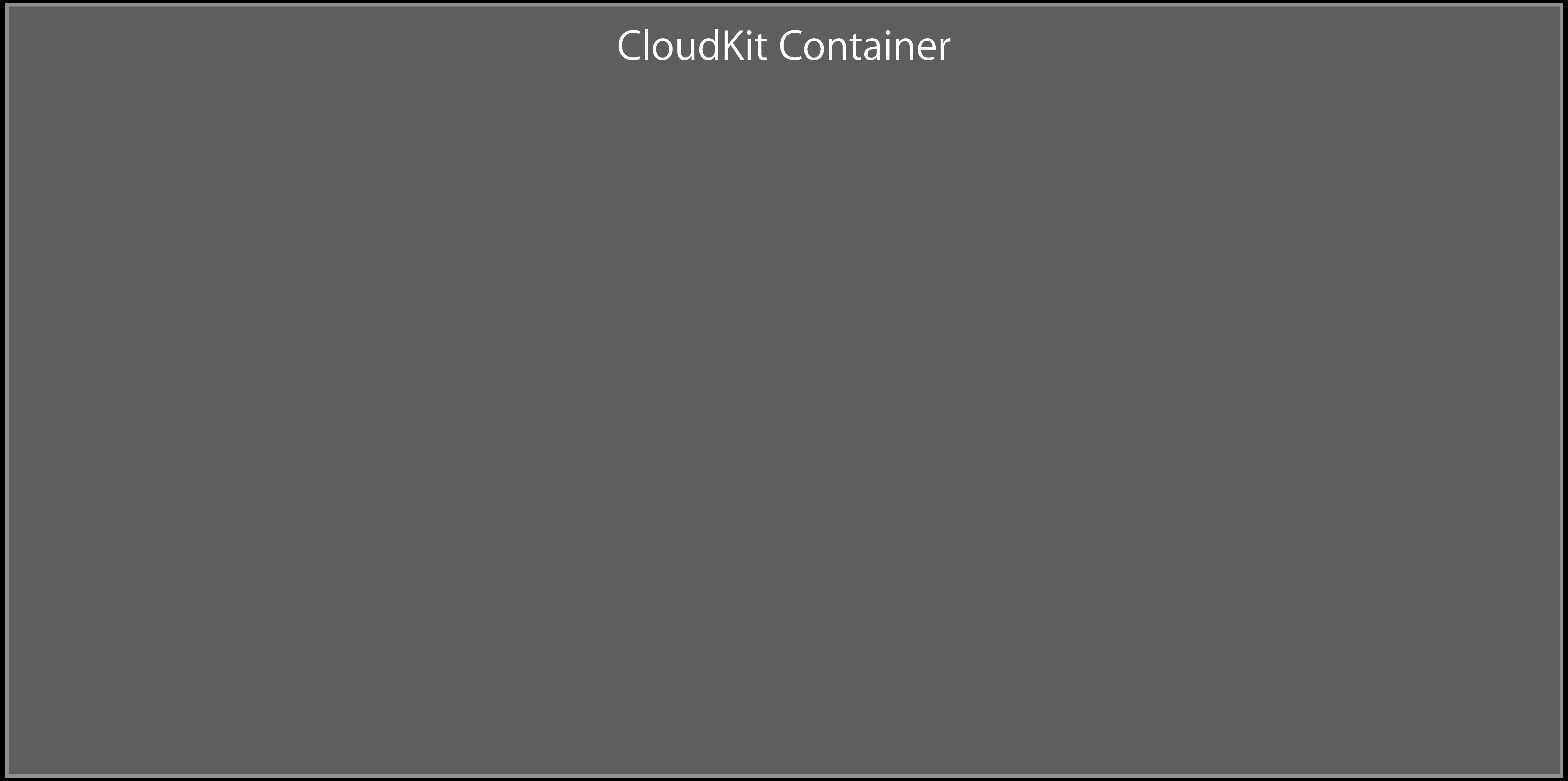
- Container
- Database
- Record
- Record Zone



CloudKit Overview

Core objects

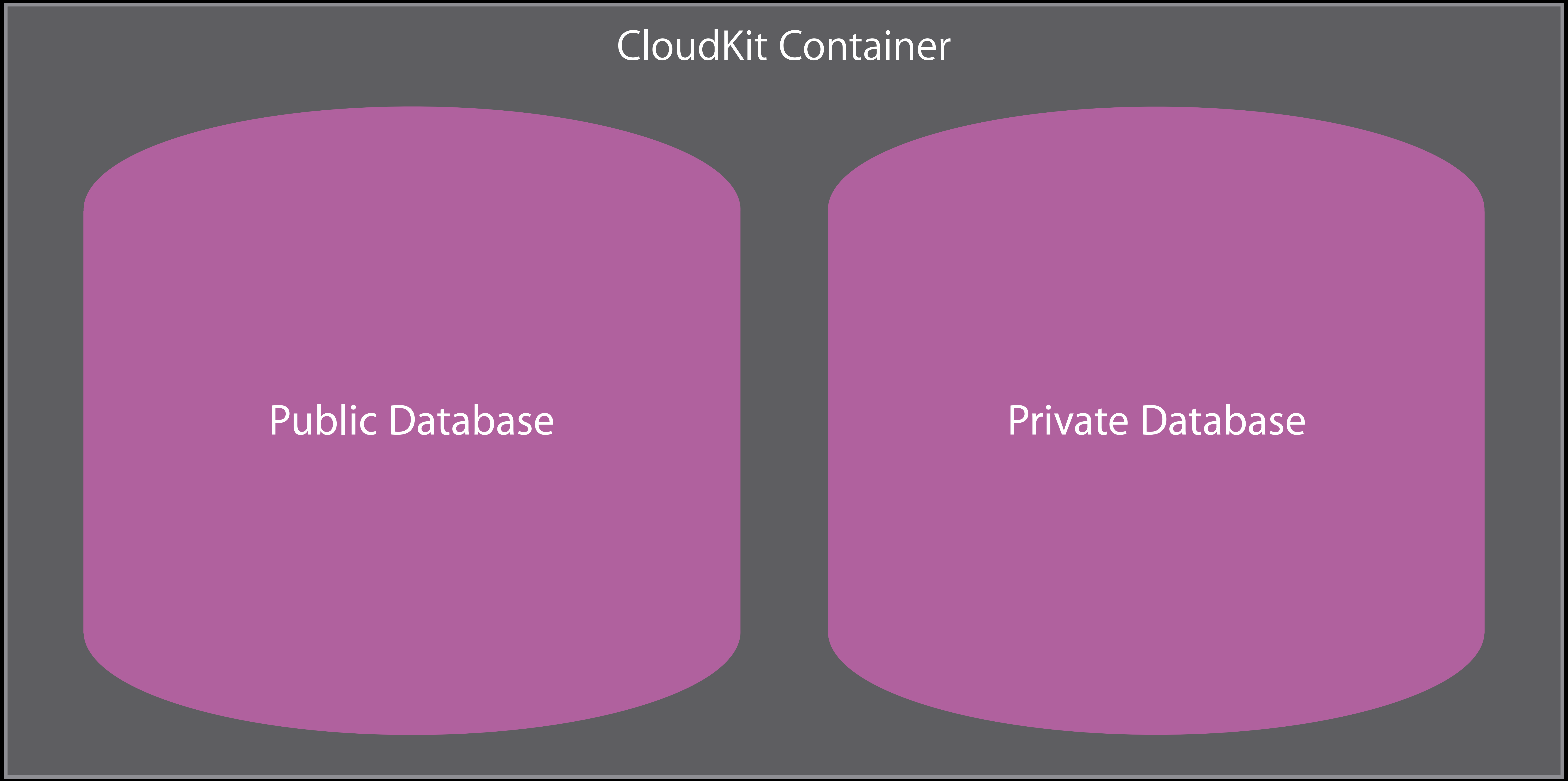
- Container
- Database
- Record
- Record Zone



CloudKit Overview

Core objects

- Container
- Database
- Record
- Record Zone

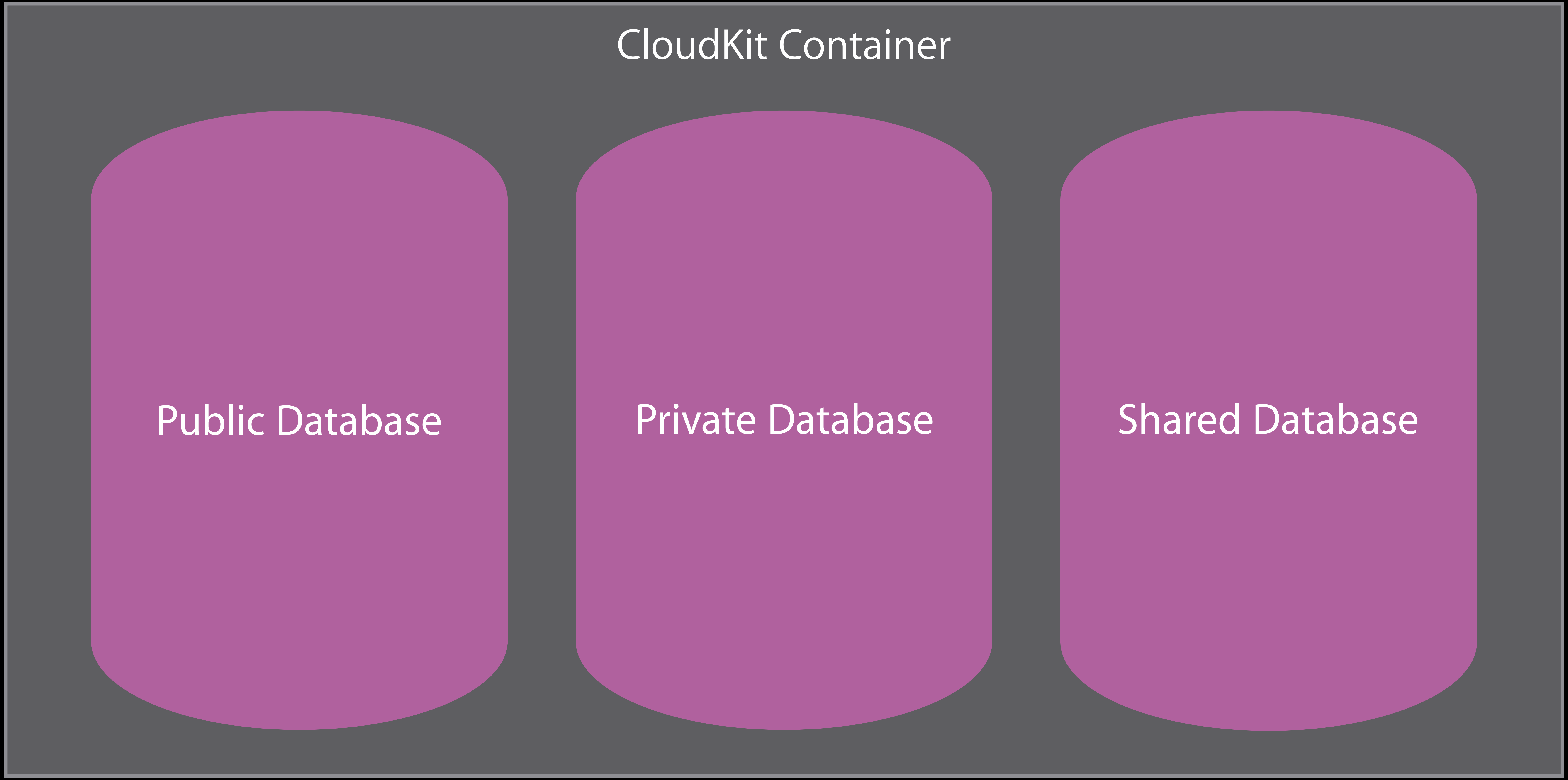


CloudKit Overview

Core objects

NEW

- Container
- Database
- Record
- Record Zone



CloudKit Overview

Core objects

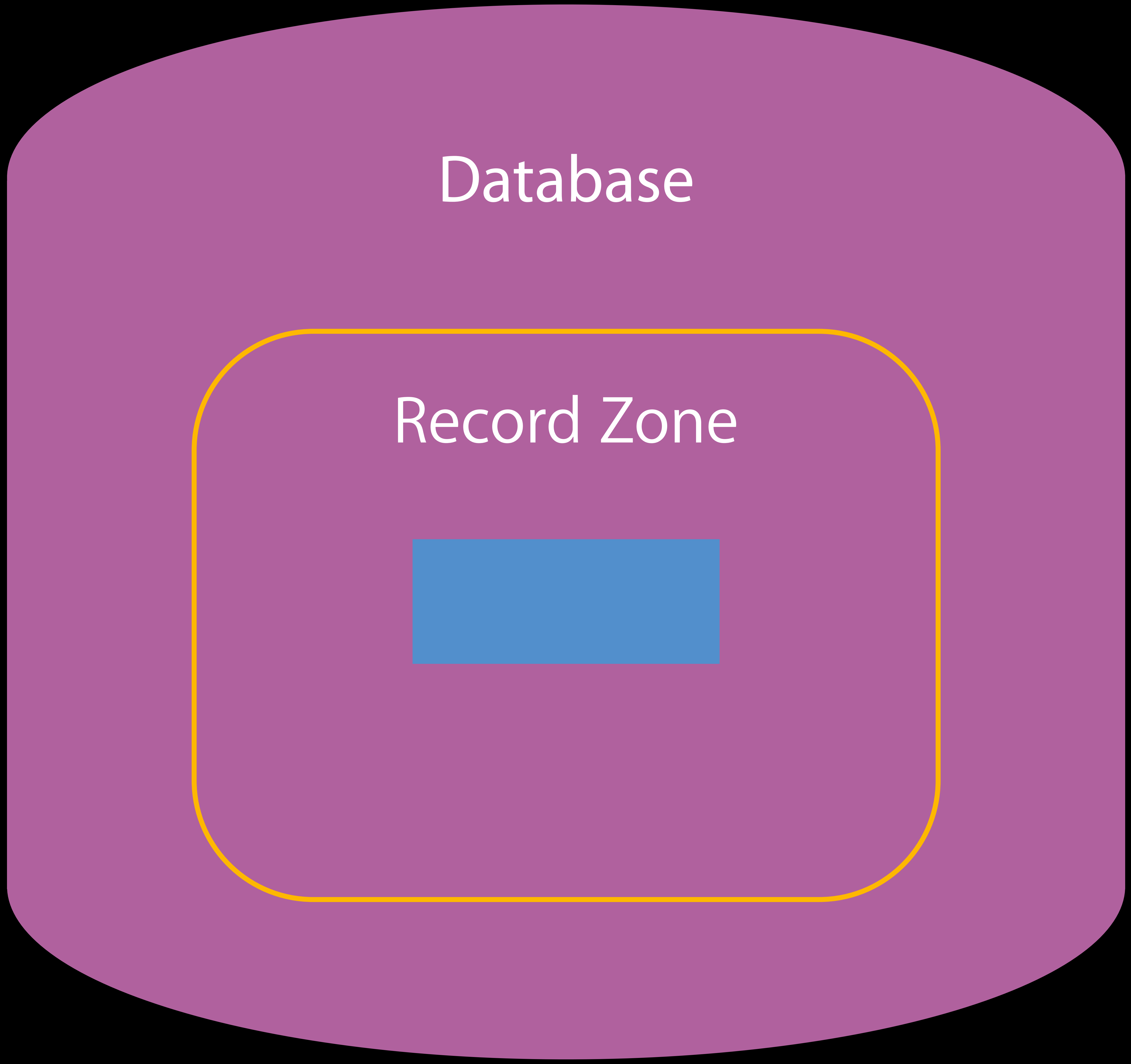
- Container
- Database
- Record
- Record Zone

Note	
Key	Value
Title	String
Body	String
CreationDate	Date
Folder	Reference

CloudKit Overview

Core objects

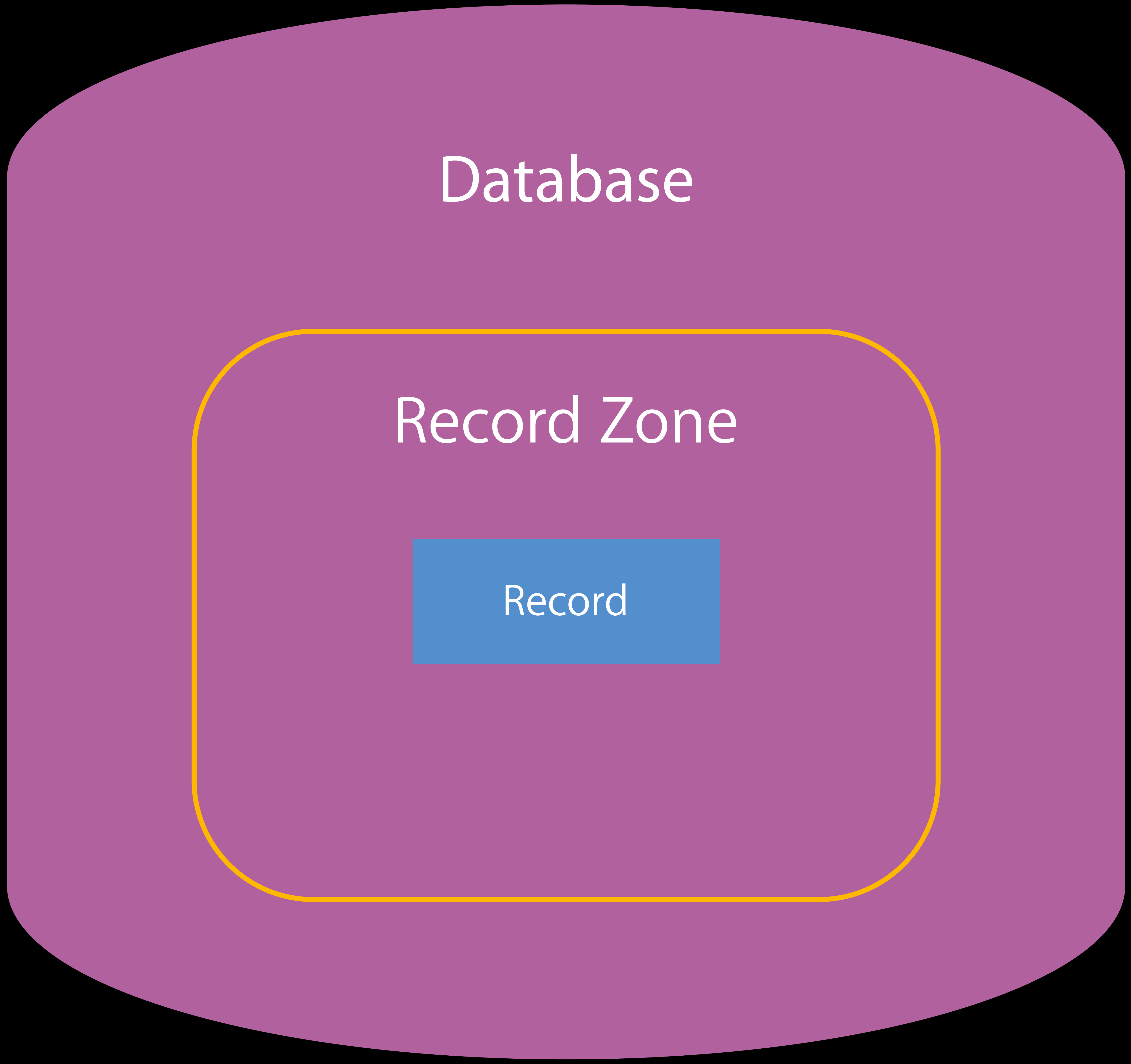
- Container
- Database
- Record
- Record Zone



CloudKit Overview

Core objects

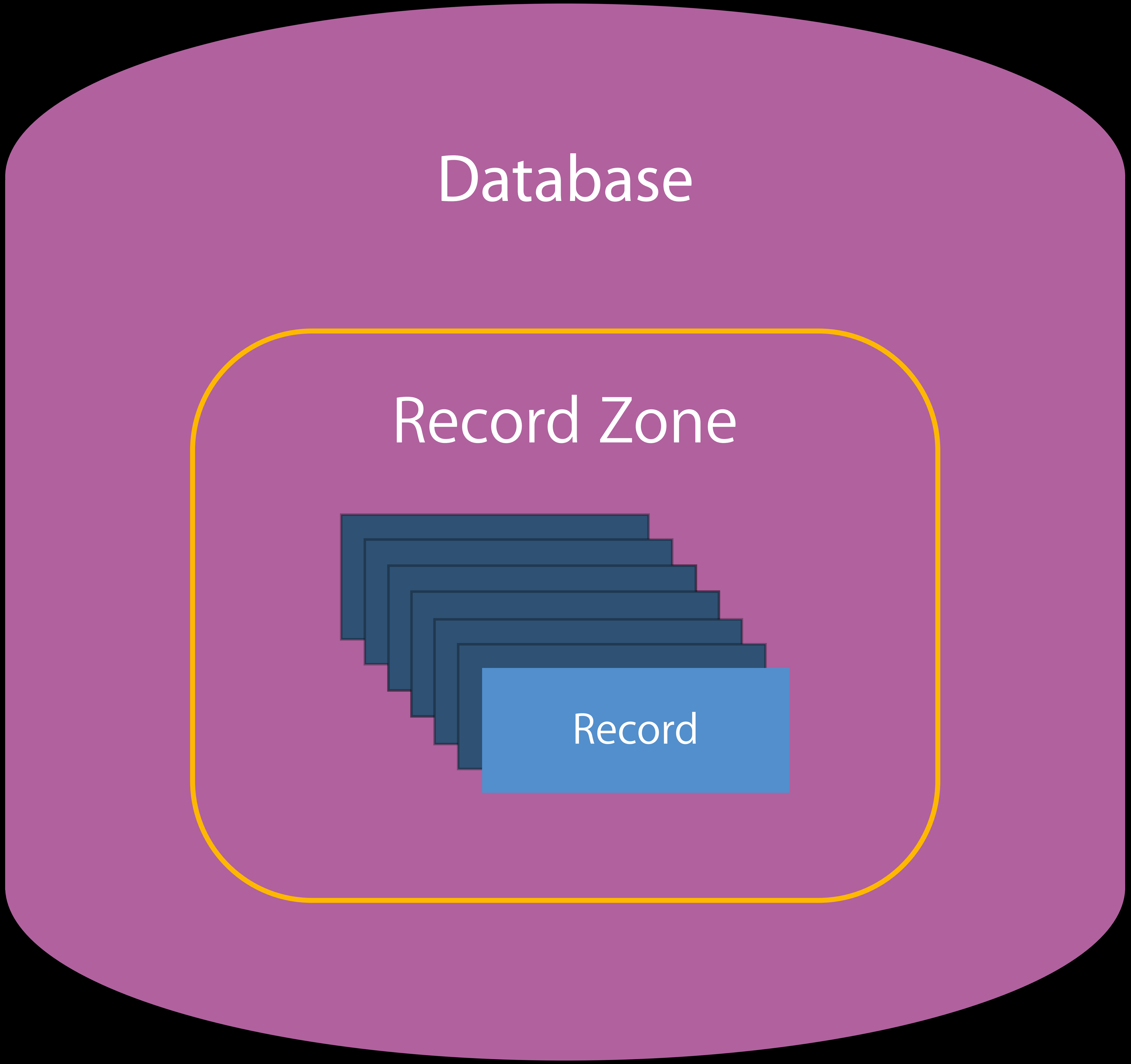
- Container
- Database
- Record
- Record Zone



CloudKit Overview

Core objects

- Container
- Database
- Record
- Record Zone



CloudKit Overview

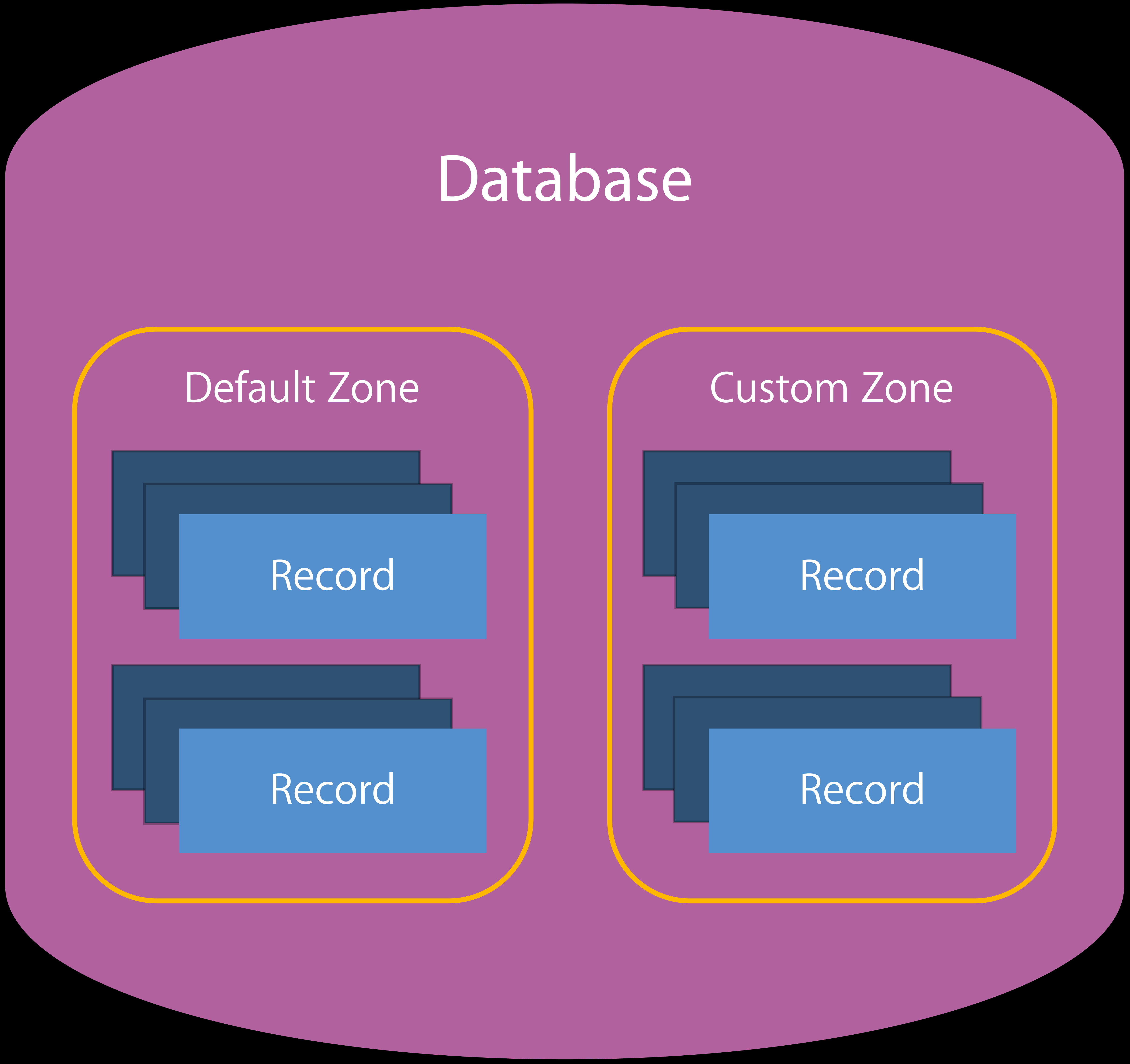
Core objects

Container

Database

Record

Record Zone



CloudKit Overview

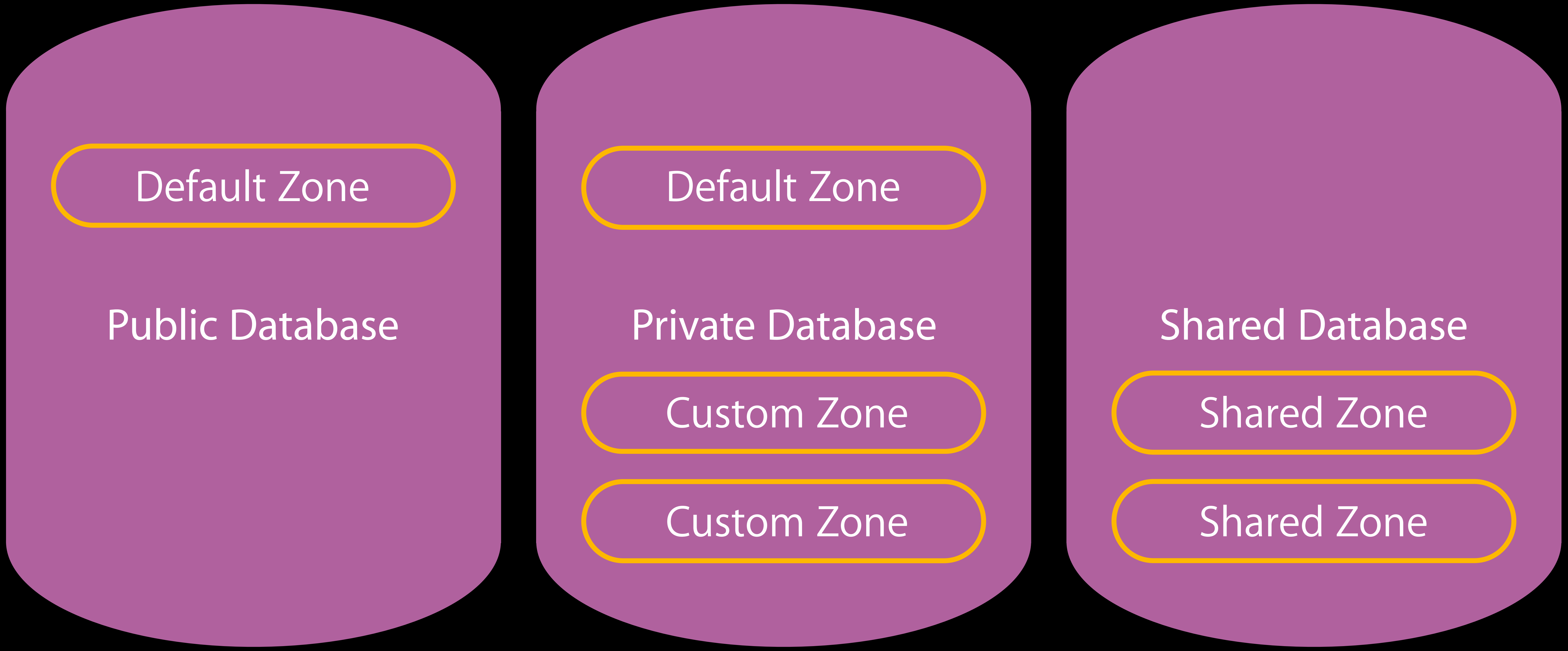
Core objects

Container

Database

Record

Record Zone



CloudKit Overview

Core objects

Container

Database

Record

Record Zone

CloudKit Overview

Core objects

NEW

Container

Database

Record

Record Zone

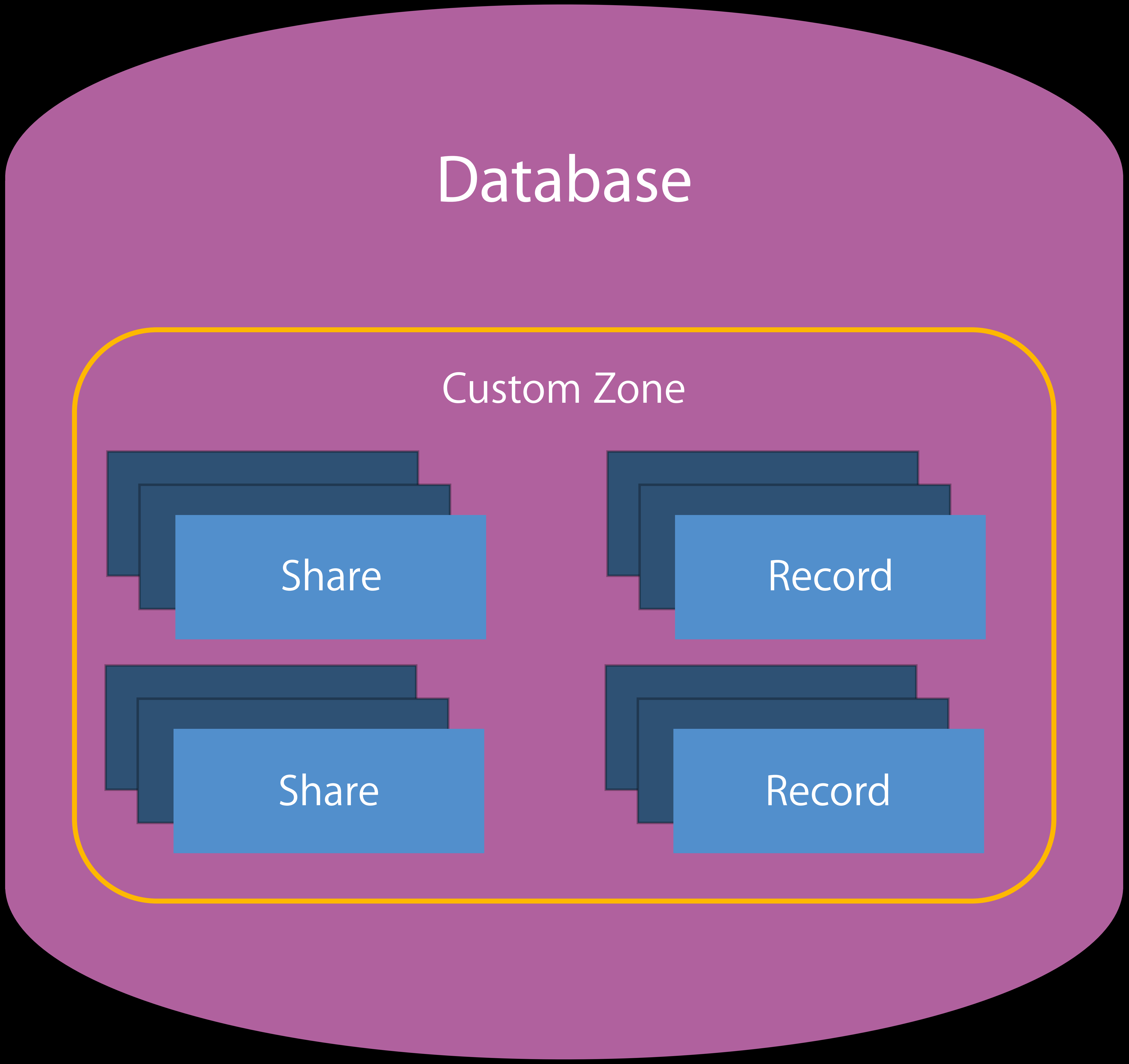
Share

CloudKit Overview

Core objects

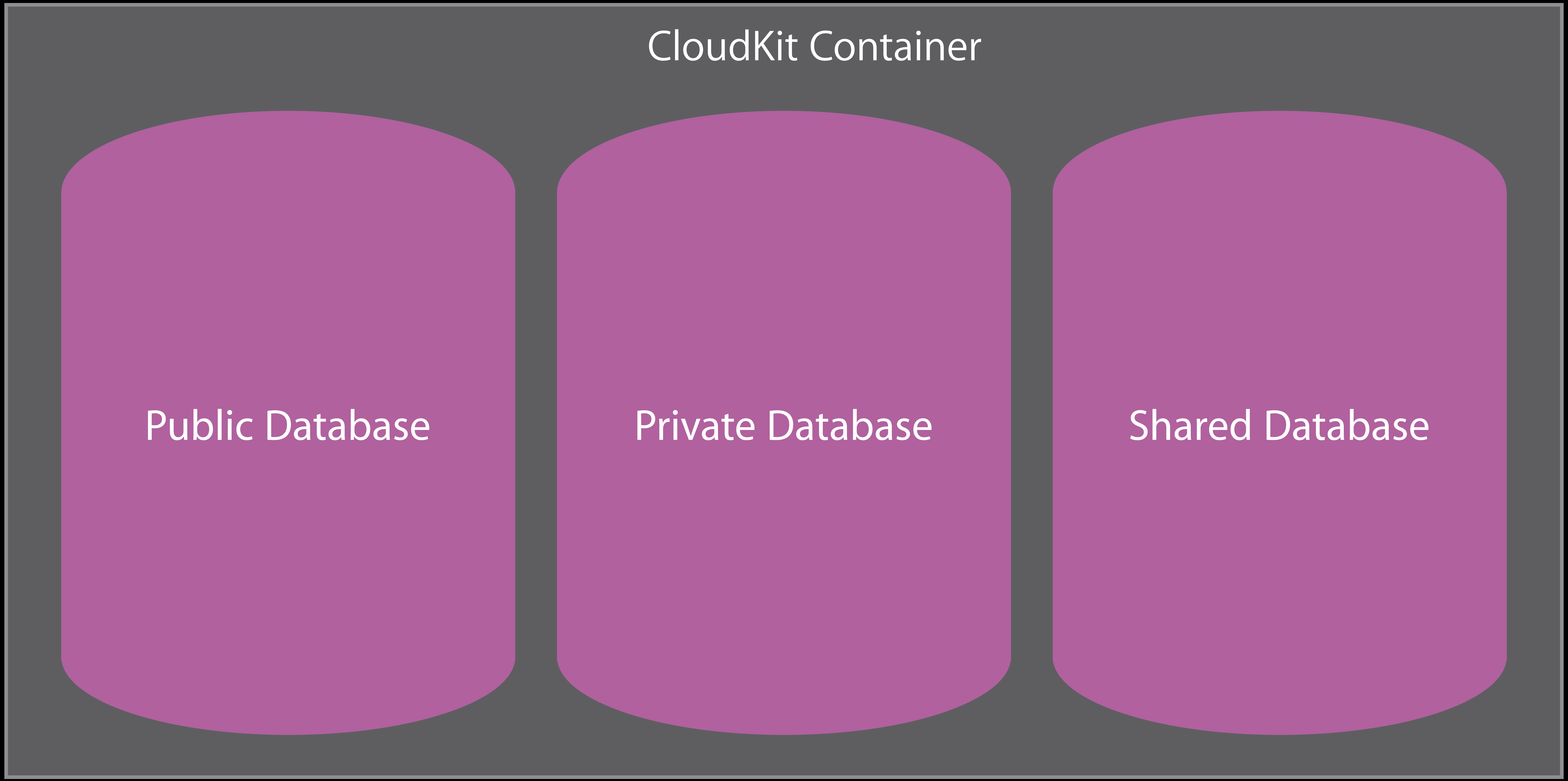
NEW

- Container
- Database
- Record
- Record Zone
- Share



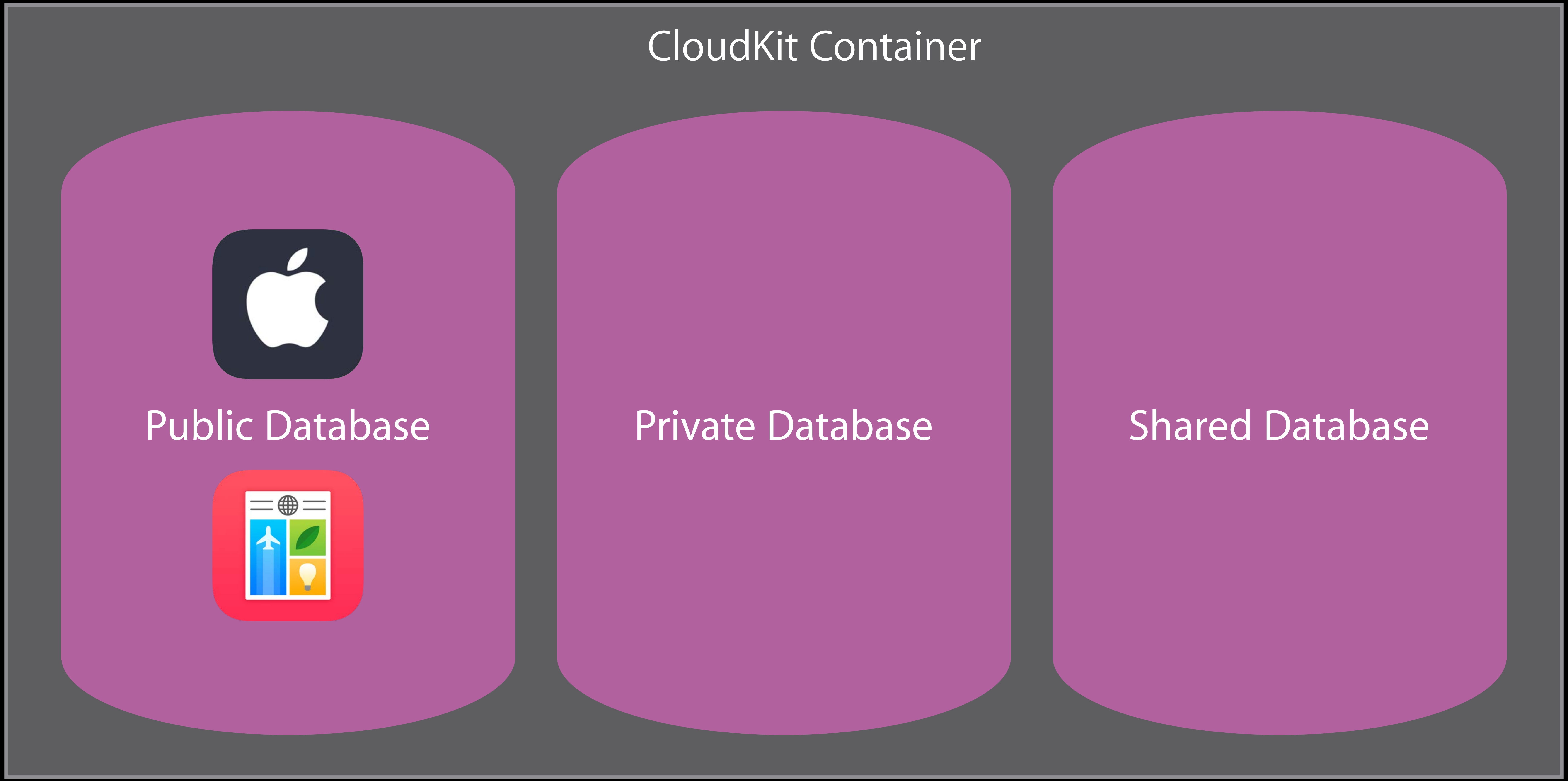
CloudKit Overview

Apple usage



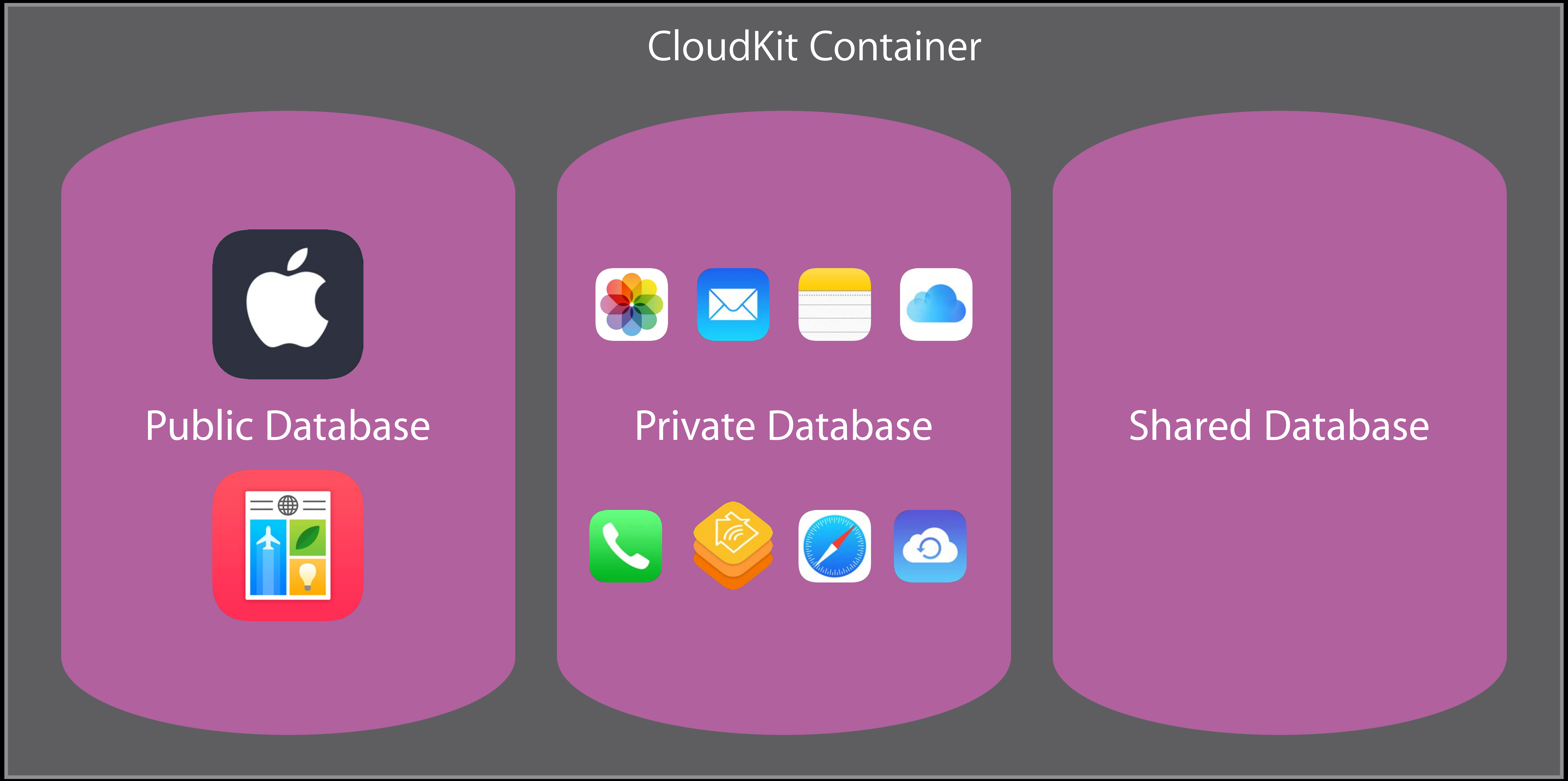
CloudKit Overview

Apple usage



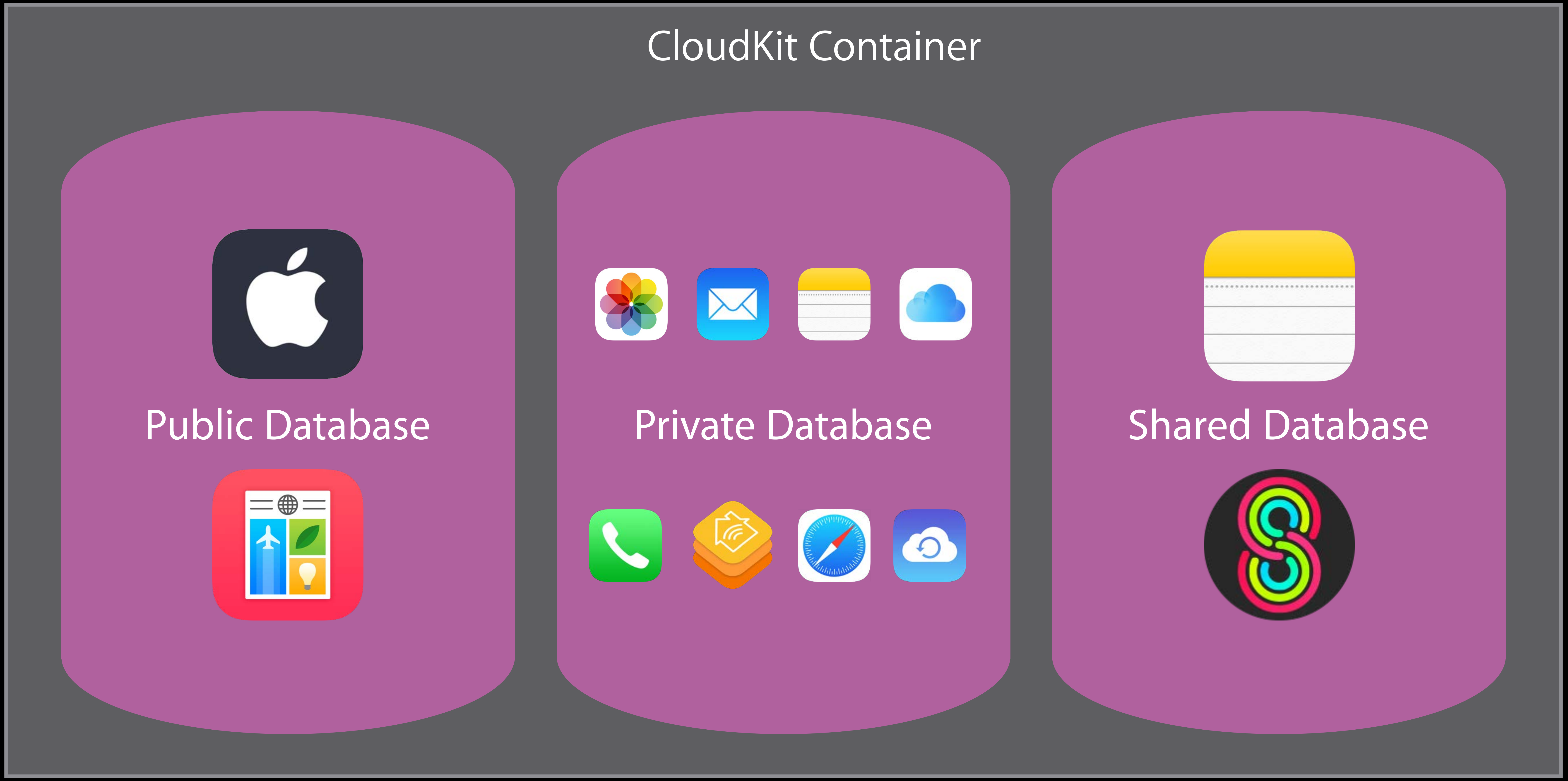
CloudKit Overview

Apple usage



CloudKit Overview

Apple usage



CloudKit Is Now Available Everywhere

CloudKit Is Now Available Everywhere

CloudKit Is Now Available Everywhere

macOS

iOS

CloudKit Is Now Available Everywhere

macOS

iOS

tvOS

CloudKit JS

Web Services

CloudKit Is Now Available Everywhere

NEW

macOS iOS

tvOS watchOS

CloudKit JS Web Services

macOS

macOS

No Mac App Store requirement

macOS

No Mac App Store requirement

- Add iCloud Capabilities via your provisioning profile

Web

Server to server

Web

Server to server

- Acts as admin user

Web

Server to server

- Acts as admin user
- Uses public/private key pair registered on CloudKit Dashboard

Web

Server to server

- Acts as admin user
- Uses public/private key pair registered on CloudKit Dashboard
- Access to public database

watchOS 3

watchOS 3

Alternative to watch connectivity code

watchOS 3

Alternative to watch connectivity code

Standalone functionality

watchOS 3

Alternative to watch connectivity code

Standalone functionality

- Can work without phone present (via wifi)

watchOS 3

Alternative to watch connectivity code

Standalone functionality

- Can work without phone present (via wifi)

Full* CloudKit API

*offer does not include CKSubscription

watchOS 3

Alternative to watch connectivity code

Standalone functionality

- Can work without phone present (via wifi)

Full* CloudKit API

Similar code on all Apple platforms

*offer does not include CKSubscription

watchOS 3

Alternative to watch connectivity code

Standalone functionality

- Can work without phone present (via wifi)

Full* CloudKit API

Similar code on all Apple platforms

Limited resources

*offer does not include CKSubscription

Telemetry

Visualize your app's behavior on the CloudKit Dashboard

NEW

Telemetry

Visualize your app's behavior on the CloudKit Dashboard

Telemetry

Visualize your app's behavior

CloudKit Dashboard

Telemetry

Visualize your app's behavior

CloudKit Dashboard

Public DB

Telemetry

Visualize your app's behavior

CloudKit Dashboard

Public DB

Aggregated Private DBs

Telemetry

Visualize your app's behavior

CloudKit Dashboard

Public DB

Aggregated Private DBs

Hour/day/week/month

Telemetry

Visualize your app's behavior

CloudKit Dashboard

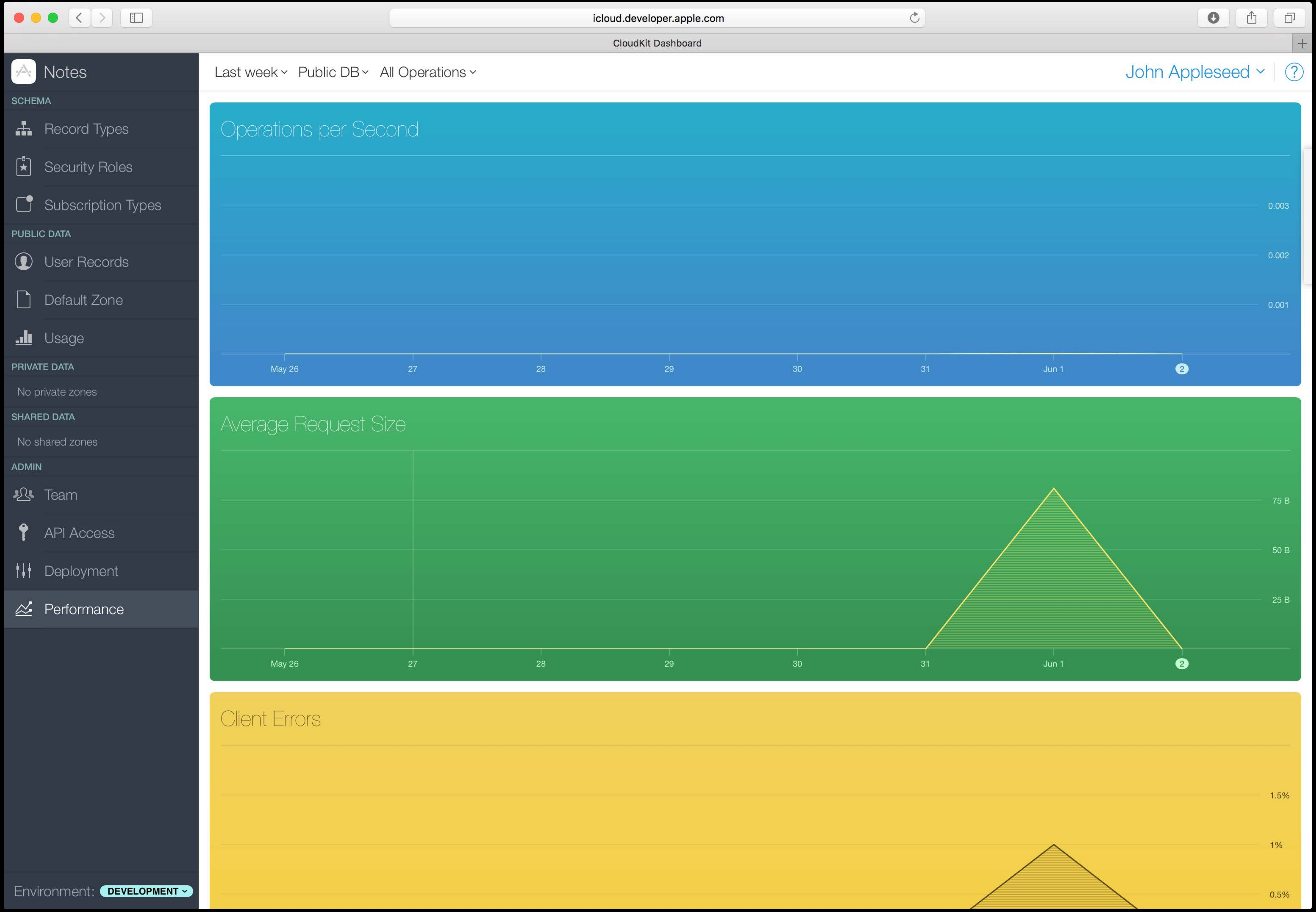
Public DB

Aggregated Private DBs

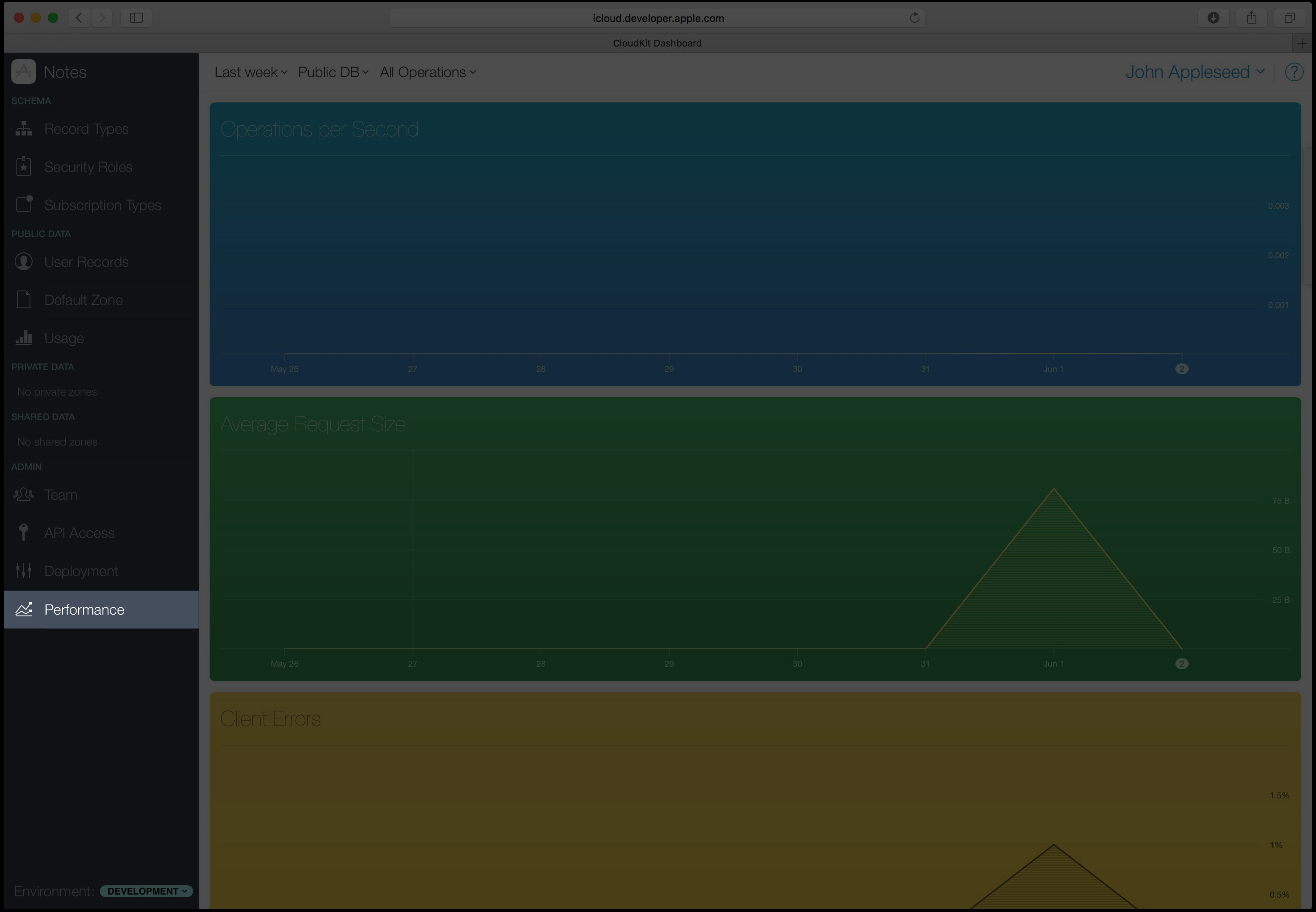
Hour/day/week/month

Per Operation type/all Operations

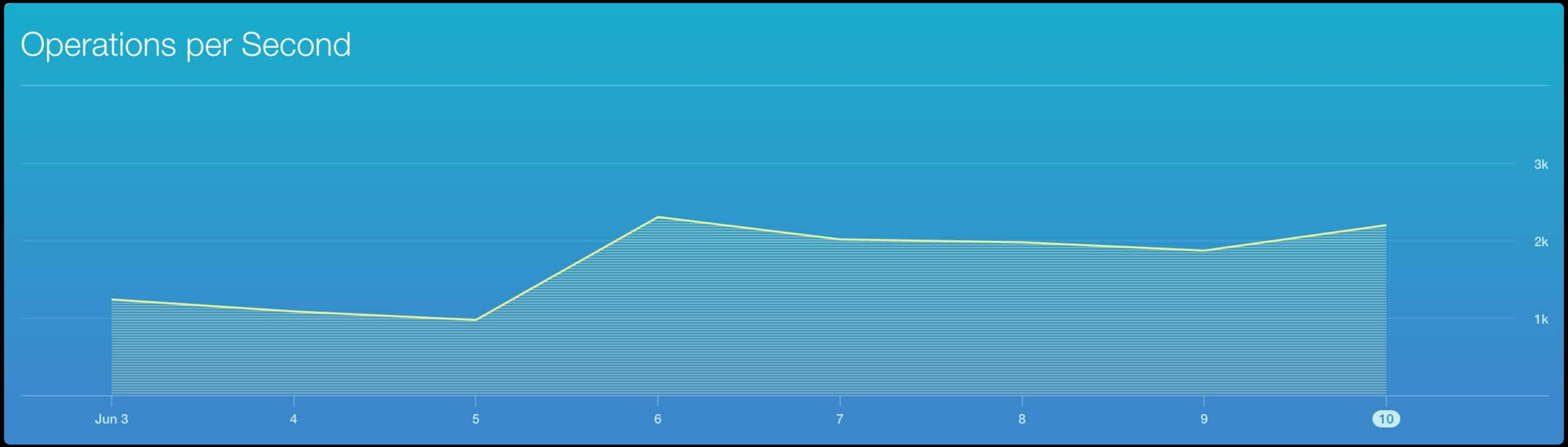
Telemetry

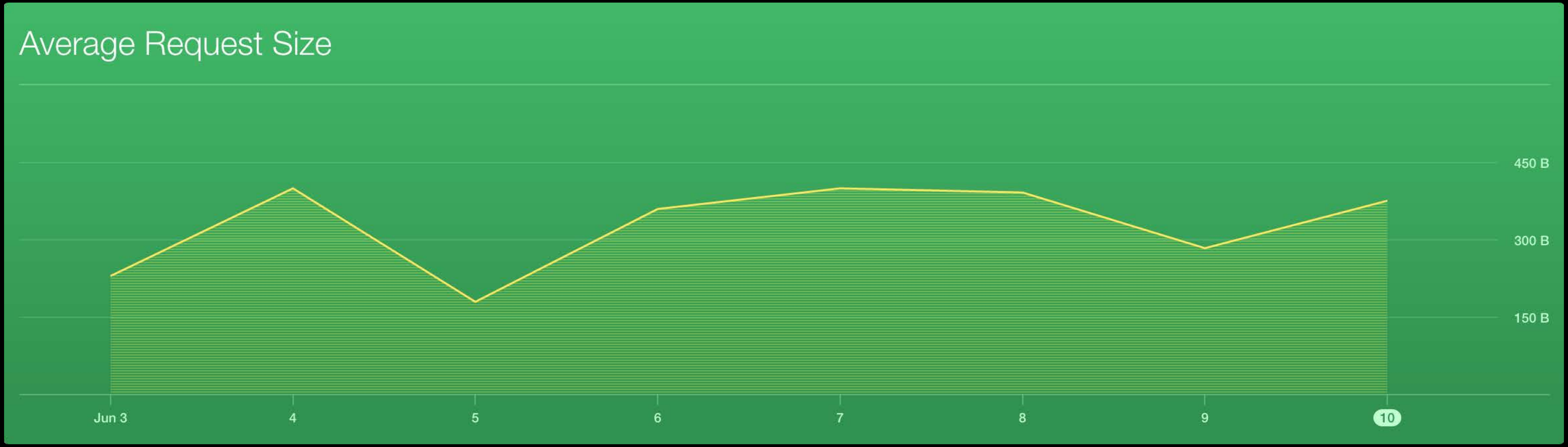

icloud.developer.apple.com
CloudKit Dashboard
Notes
SCHEMA
Record Types
Security Roles
Subscription Types
PUBLIC DATA
User Records
Default Zone
Usage
PRIVATE DATA
No private zones
SHARED DATA
No shared zones
ADMIN
Team
API Access
Deployment
Performance
Environment: DEVELOPMENT
Last week
Public DB
All Operations
John Appleseed
Operations per Second
0.003
0.002
0.001
May 26
27
28
29
30
31
Jun 1
2
Average Request Size
75 B
50 B
25 B
Client Errors
1.5%
1%
0.5%

Telemetry

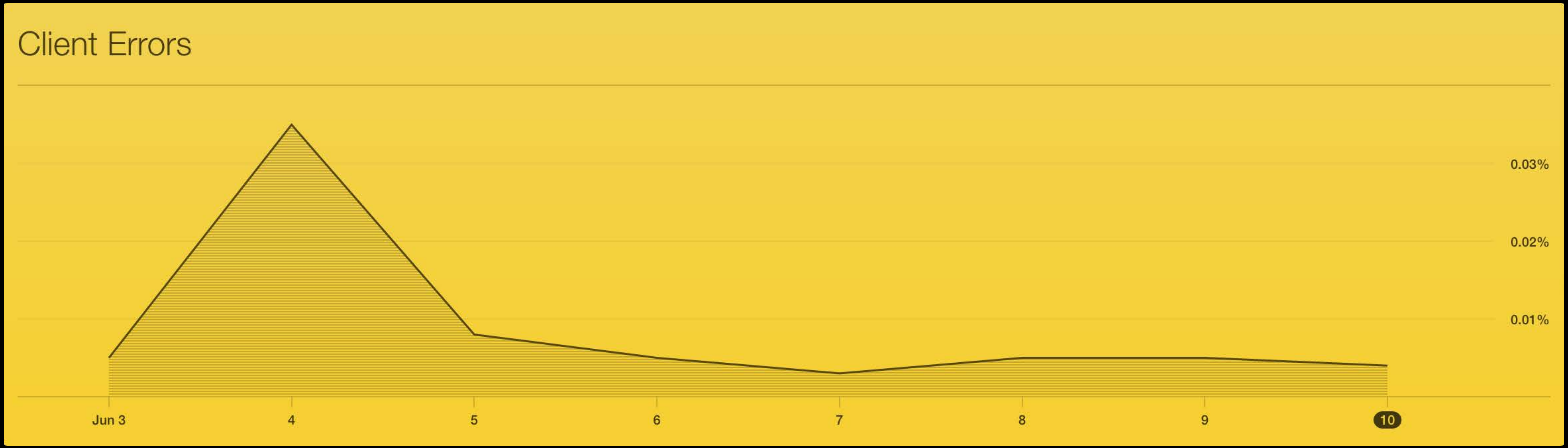

icloud.developer.apple.com
CloudKit Dashboard
Notes
SCHEMA
Record Types
Security Roles
Subscription Types
PUBLIC DATA
User Records
Default Zone
Usage
PRIVATE DATA
No private zones
SHARED DATA
No shared zones
ADMIN
Team
API Access
Deployment
Performance
Environment: DEVELOPMENT
Last week
Public DB
All Operations
John Appleseed
Operations per Second
0.003
0.002
0.001
May 26
27
28
29
30
31
Jun 1
2
Average Request Size
75 B
50 B
25 B
Client Errors
1.5%
1%
0.5%

Performance

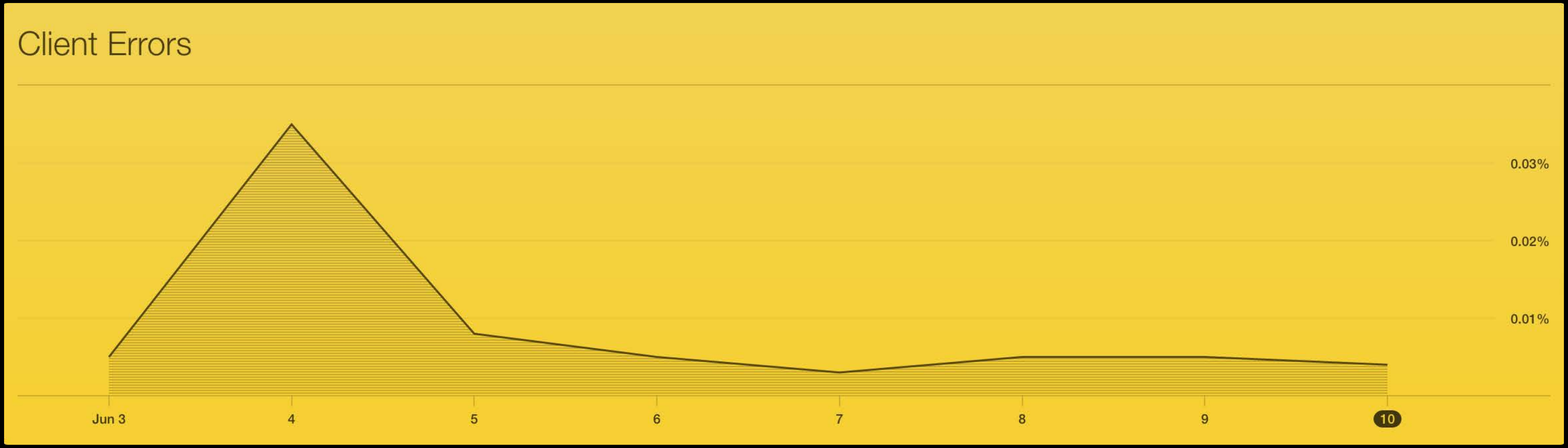




Correctness

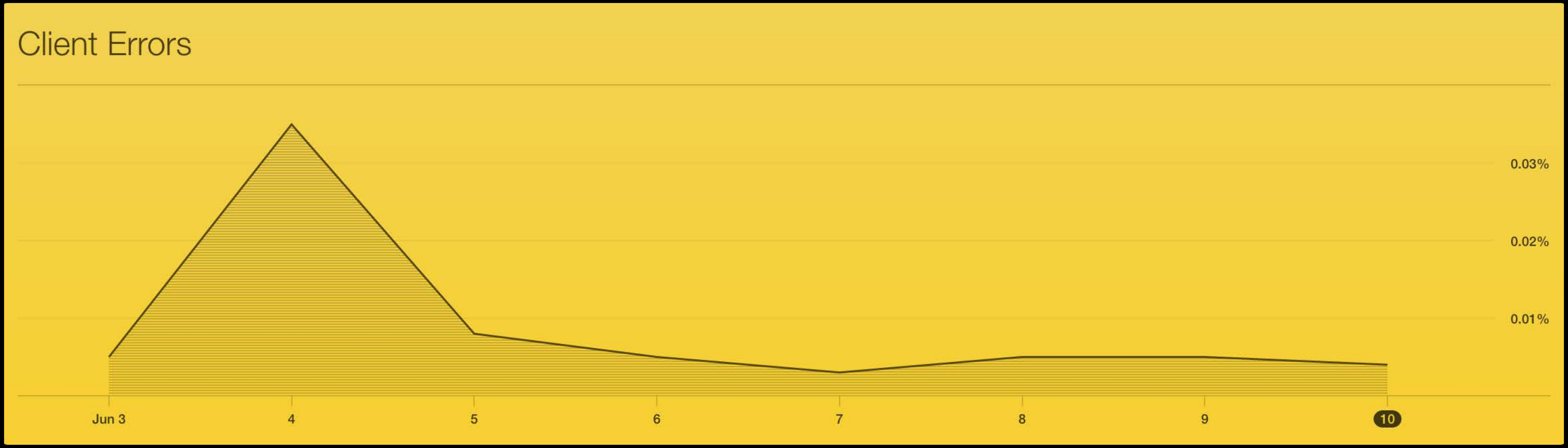


Correctness



Detect client changes with abnormally frequent errors

Correctness



Detect client changes with abnormally frequent errors

Error handling is essential

API Improvements

NEW

API Improvements

API Improvements

Long Lived Operations

CKOperation Timeouts

Handling Many Record Zones

Fetching Multiple Change Sets

API Improvements

Long Lived Operations

CKOperation Timeouts

Handling Many Record Zones

Fetching Multiple Change Sets

Long-Lived Operations

Don’t repeat your work

Long-Lived Operations

Don’t repeat your work

Operations keep running if your application exits

Long-Lived Operations

Don’t repeat your work

Operations keep running if your application exits

Callbacks saved by CloudKit

Long-Lived Operations

Don’t repeat your work

Operations keep running if your application exits

Callbacks saved by CloudKit

Operation can be replayed later

API Improvements

Long Lived Operations

CKOperation Timeouts

Handling Many Record Zones

Fetching Multiple Change Sets

API Improvements

Long Lived Operations

CKOperation Timeouts

Handling Many Record Zones

Fetching Multiple Change Sets

CKOperation Timeouts

How long do you want to wait?

QualityOfService	Behavior on broken network
userInteractive	
userInitiated	
utility	
background	
default	

CKOperation Timeouts

How long do you want to wait?

QualityOfService	Behavior on broken network
userInteractive	timeout after 60 seconds
userInitiated	timeout after 60 seconds
utility	
background	
default	

CKOperation Timeouts

How long do you want to wait?

QualityOfService	Behavior on broken network
userInteractive	timeout after 60 seconds
userInitiated	timeout after 60 seconds
utility	timeout after 7 days
background	timeout after 7 days
default	timeout after 7 days

CKOperation Timeouts

How long do you want to wait?

QualityOfService	Behavior on broken network
userInteractive	timeout after 60 seconds
userInitiated	timeout after 60 seconds
utility	timeout after 7 days
background	timeout after 7 days
default	timeout after 7 days

CKOperation Timeouts

How long do you want to wait?

QualityOfService	Behavior on broken network
userInteractive	timeout after 60 seconds
userInitiated	timeout after 60 seconds
utility	timeout after 7 days
background	timeout after 7 days
default	timeout after 7 days

CKOperation Timeouts

How long do you want to wait?

CKOperation Timeouts

How long do you want to wait?

Network inactivity

- Use the `timeoutIntervalForRequest` property on CKOperation
- Default value is 60 seconds

CKOperation Timeouts

How long do you want to wait?

Network inactivity

- Use the `timeoutIntervalForRequest` property on CKOperation
- Default value is 60 seconds

Start-to-finish timeout

- Use the `timeoutIntervalForResource` property on CKOperation
- Default value is 7 days
- CKOperation may stay around longer

API Improvements

Long Lived Operations

CKOperation Timeouts

Handling Many Record Zones

Fetching Multiple Change Sets

API Improvements

Long Lived Operations

CKOperation Timeouts

Handling Many Record Zones

Fetching Multiple Change Sets

Handling Many Record Zones

Reduce payloads and roundtrips

CKFetchRecordZonesOperation

- Poll for all record zones in a database

Handling Many Record Zones

Reduce payloads and roundtrips

✗

CKFetchRecordZonesOperation

- Poll for all record zones in a database

Handling Many Record Zones

Reduce payloads and roundtrips

CKFetchRecordZonesOperation

- Poll for all record zones in a database

Handling Many Record Zones

Reduce payloads and roundtrips

✓

CKDatabaseSubscription

- Receive a push for each change in a database

CKFetchDatabaseChangesOperation

- Fetch ids of record zones with changes

✗

CKFetchRecordZonesOperation

- Poll for all record zones in a database

Handling Many Record Zones

Reduce payloads and roundtrips

✓

CKDatabaseSubscription

- Receive a push for each change in a database

CKFetchDatabaseChangesOperation

- Fetch ids of record zones with changes

✕

CKFetchRecordZonesOperation

- Poll for all record zones in a database

CKFetchRecordChangesOperation

- Track and fetch record changes on a per-record-zone basis

Handling Many Record Zones

Reduce payloads and roundtrips

✓

CKDatabaseSubscription

- Receive a push for each change in a database

CKFetchDatabaseChangesOperation

- Fetch ids of record zones with changes

✗

CKFetchRecordZonesOperation

- Poll for all record zones in a database

✗

CKFetchRecordChangesOperation

- Track and fetch record changes on a per-record-zone basis

Handling Many Record Zones

Reduce payloads and roundtrips

✓

CKDatabaseSubscription

- Receive a push for each change in a database

CKFetchDatabaseChangesOperation

- Fetch ids of record zones with changes

✗

CKFetchRecordZonesOperation

- Poll for all record zones in a database

✗

CKFetchRecordChangesOperation

- Track and fetch record changes on a per-record-zone basis

Handling Many Record Zones

Reduce payloads and roundtrips

✓

CKDatabaseSubscription

- Receive a push for each change in a database

CKFetchDatabaseChangesOperation

- Fetch ids of record zones with changes

✗

CKFetchRecordZonesOperation

- Poll for all record zones in a database

✓

CKFetchRecordZoneChangesOperation

- Fetch record changes over multiple record zones in a single operation

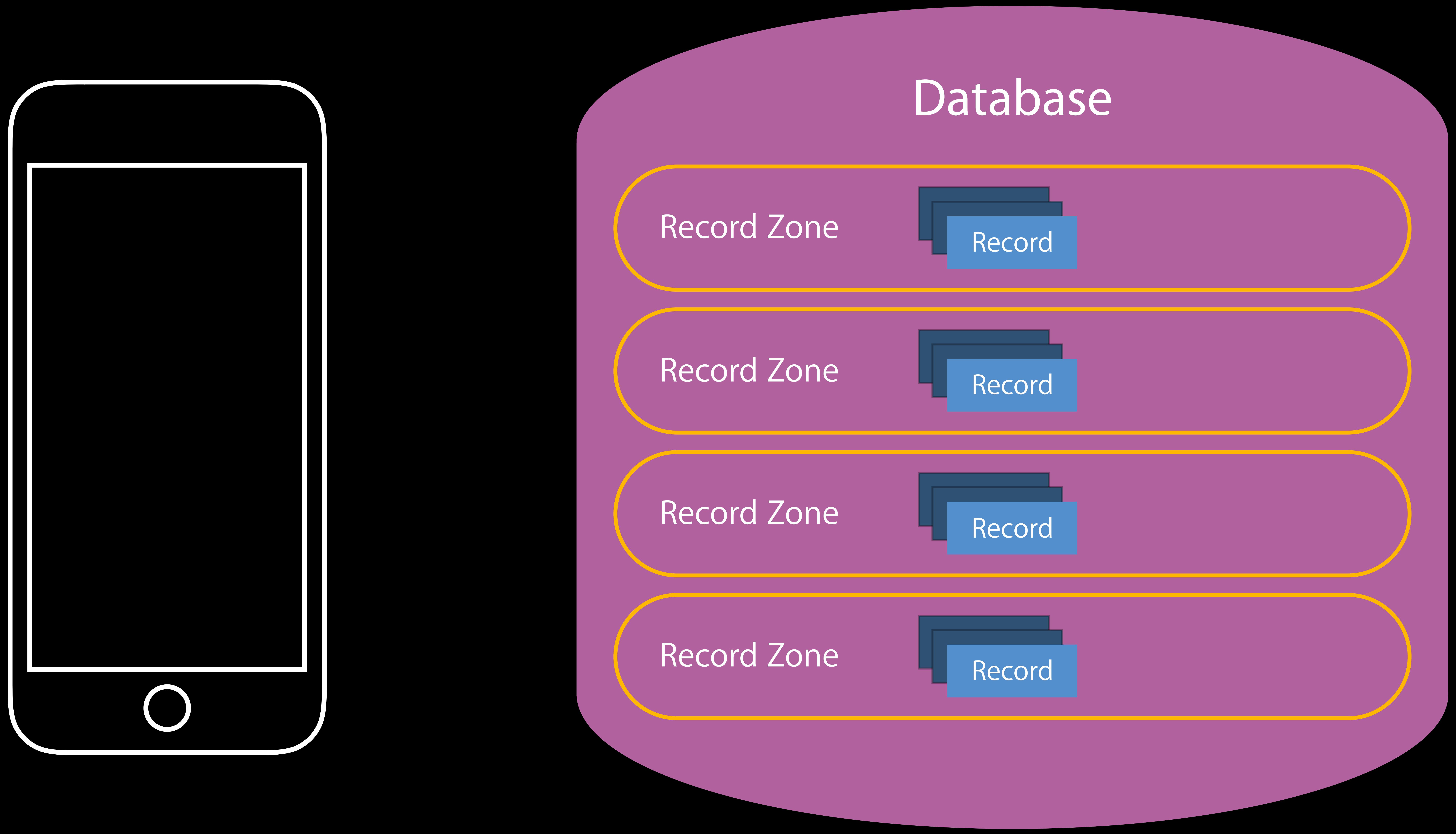
✗

CKFetchRecordChangesOperation

- Track and fetch record changes on a per-record-zone basis

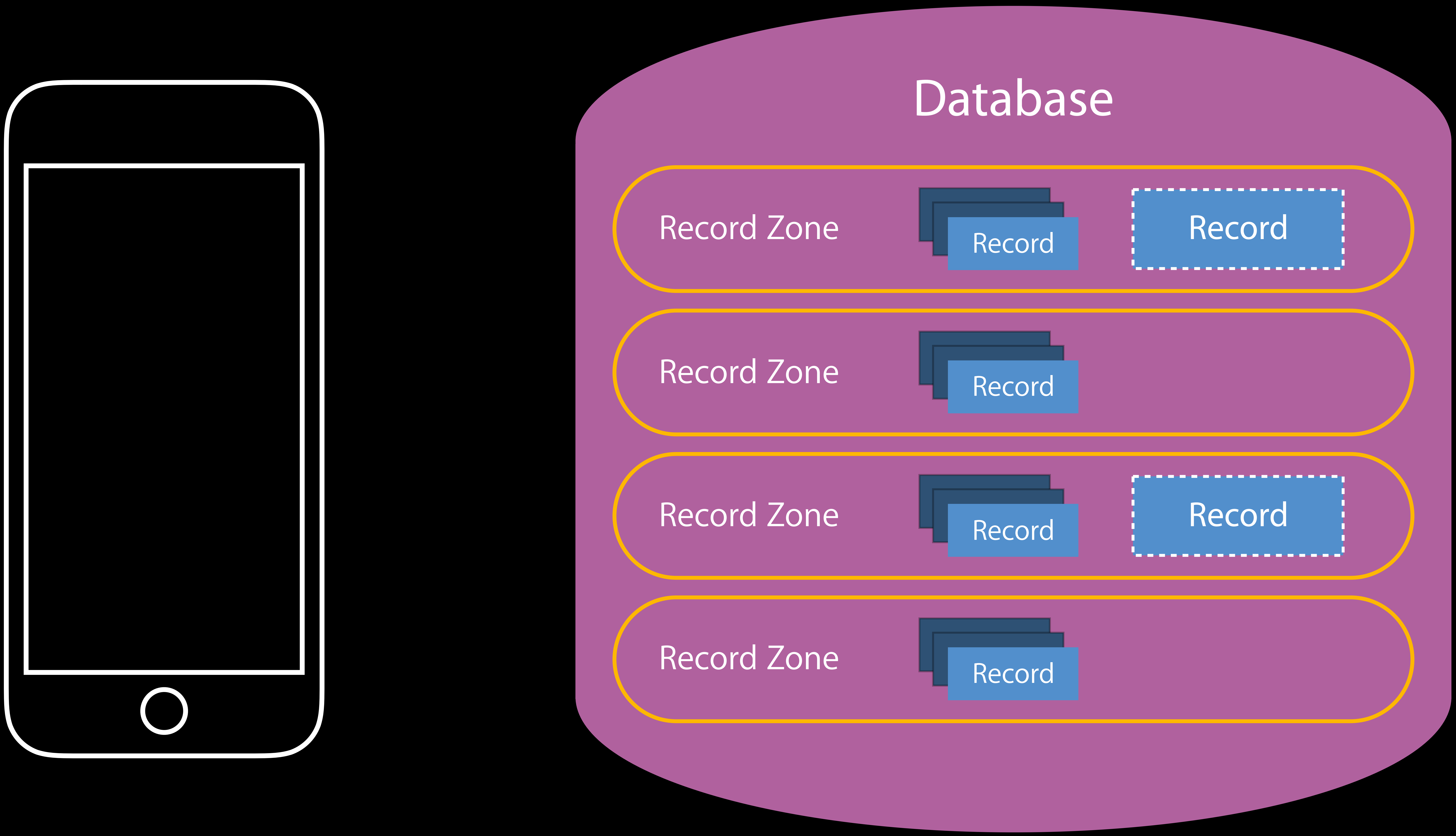
Handling Many Record Zones

CKDatabaseSubscription



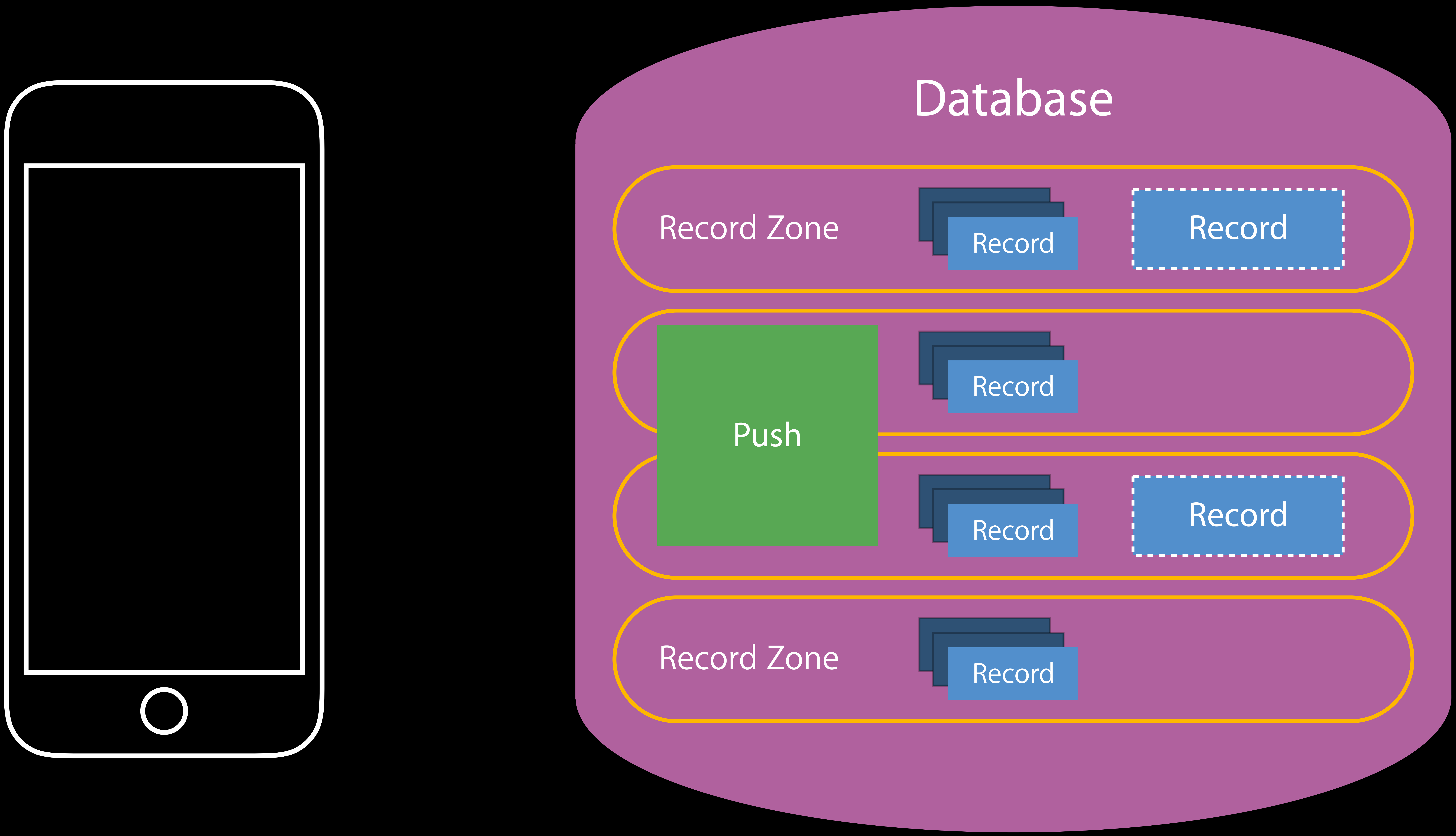
Handling Many Record Zones

CKDatabaseSubscription



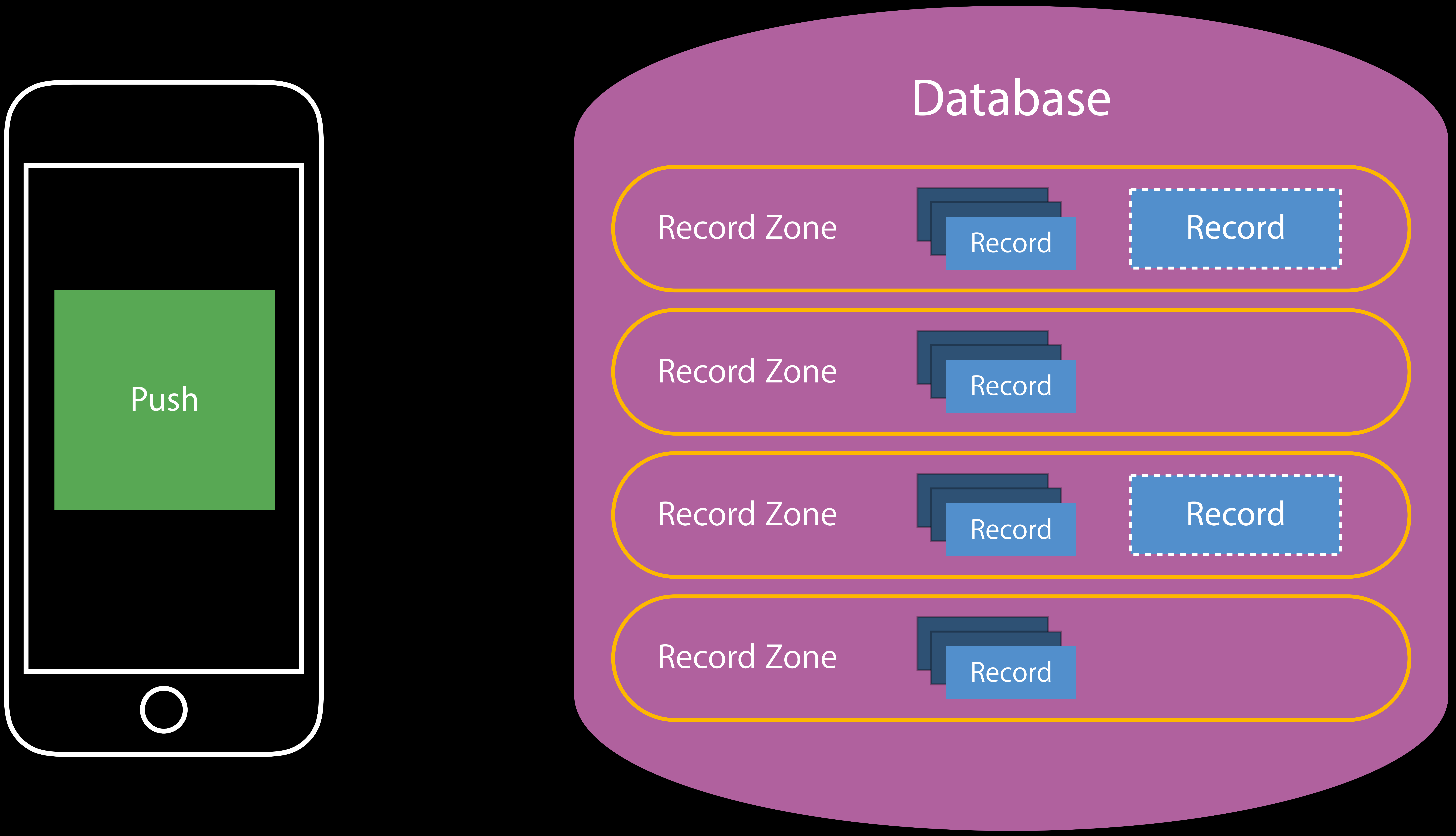
Handling Many Record Zones

CKDatabaseSubscription



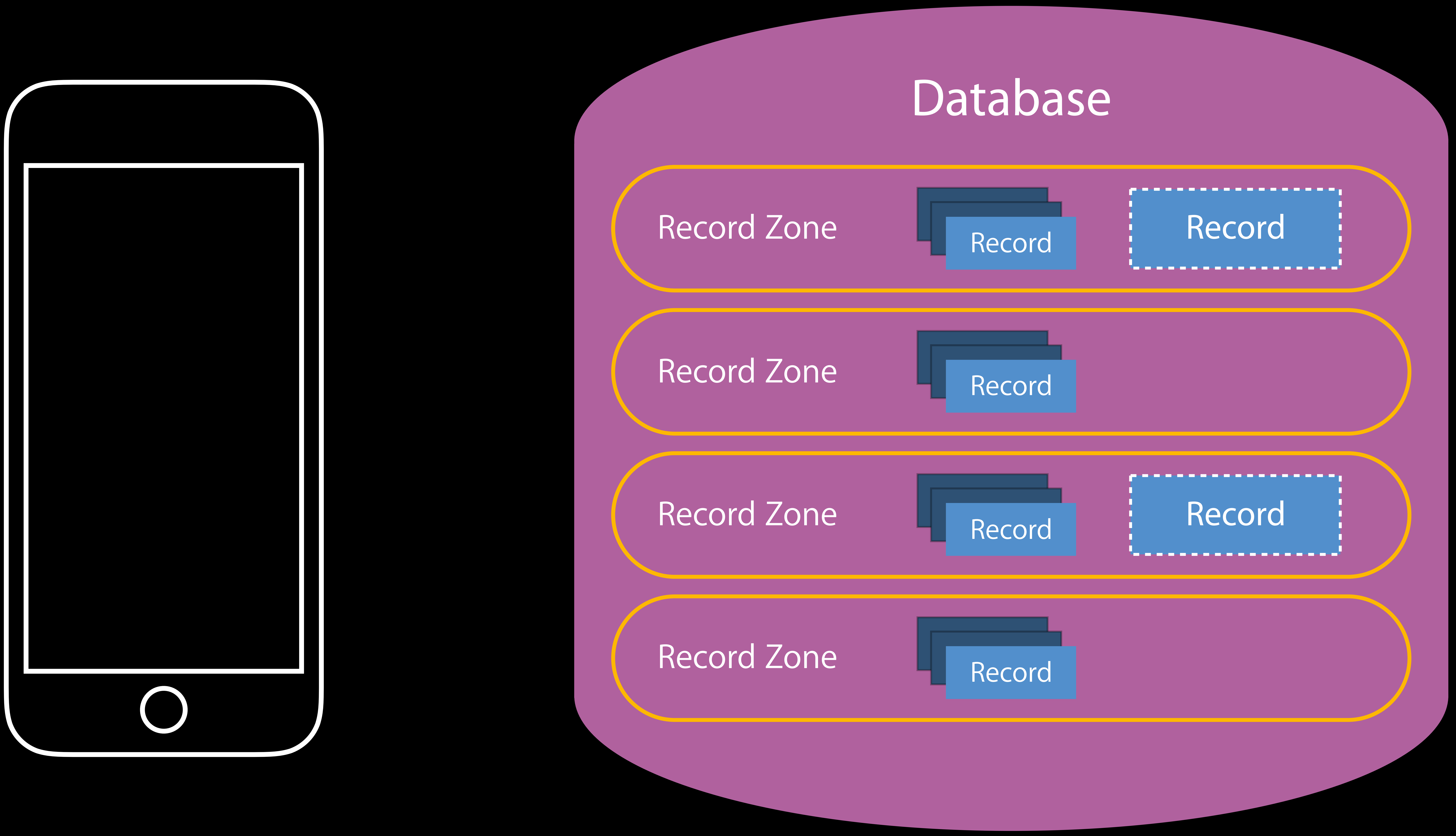
Handling Many Record Zones

CKDatabaseSubscription



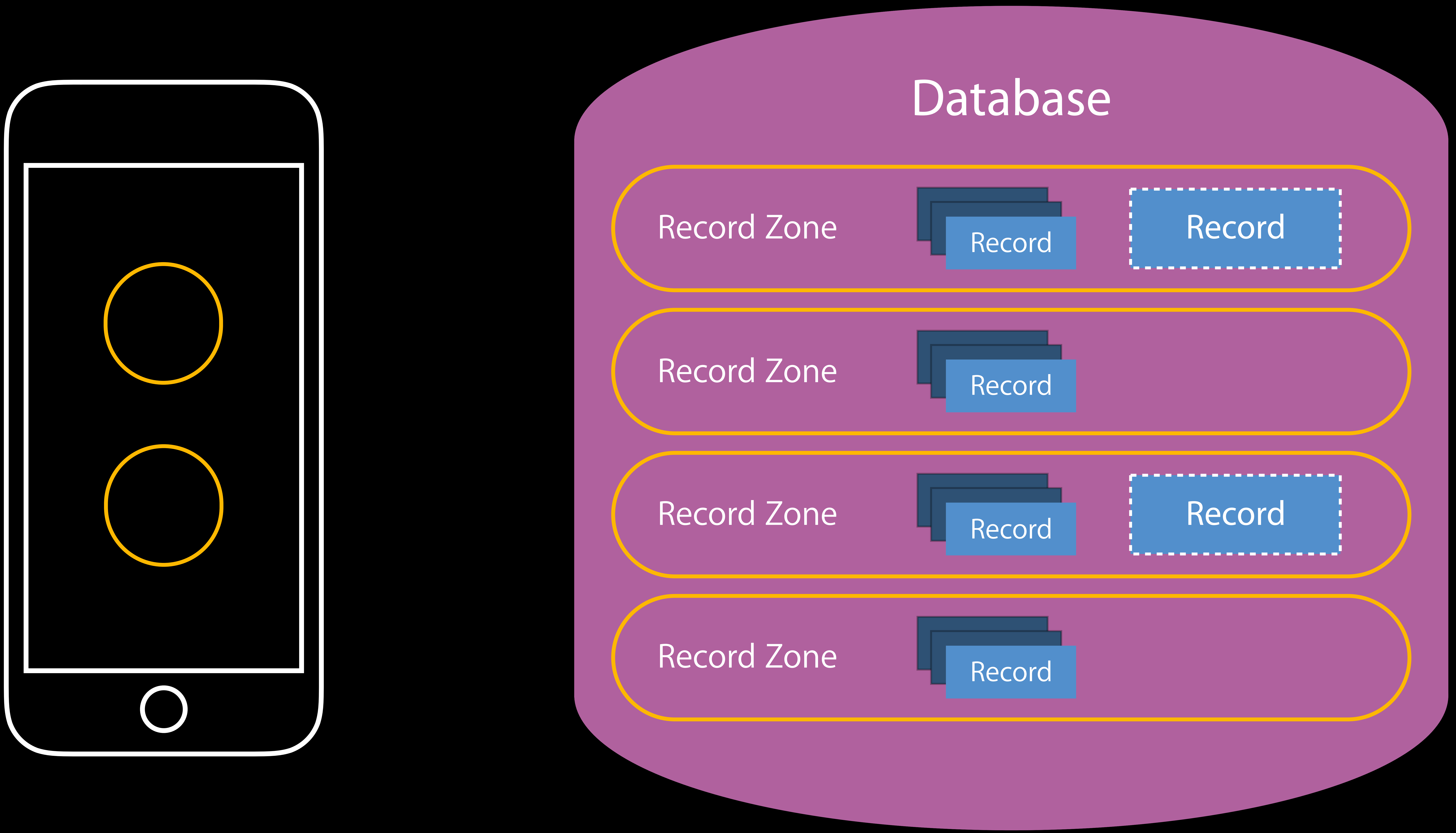
Handling Many Record Zones

CKFetchDatabaseChangesOperation



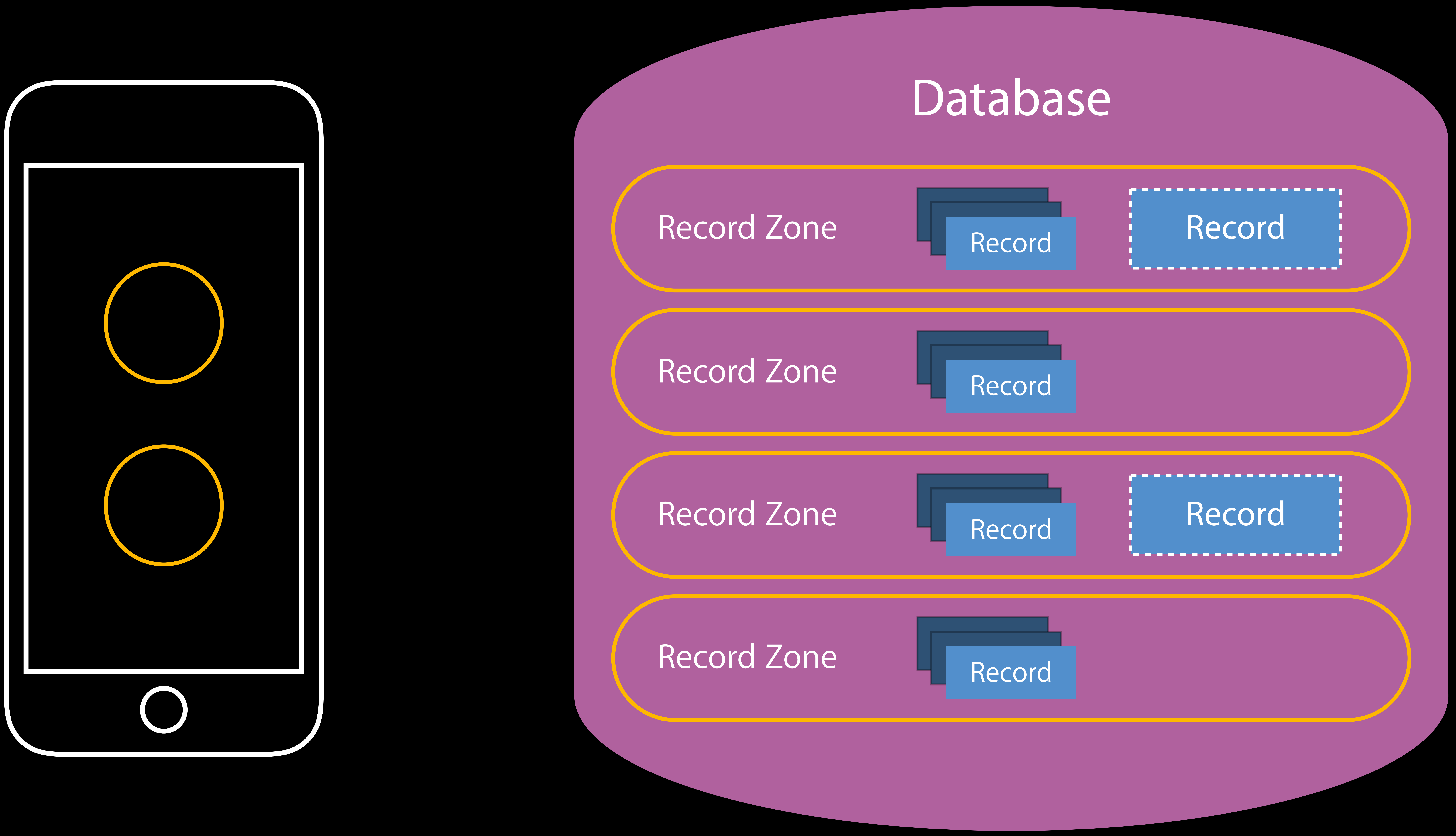
Handling Many Record Zones

CKFetchDatabaseChangesOperation



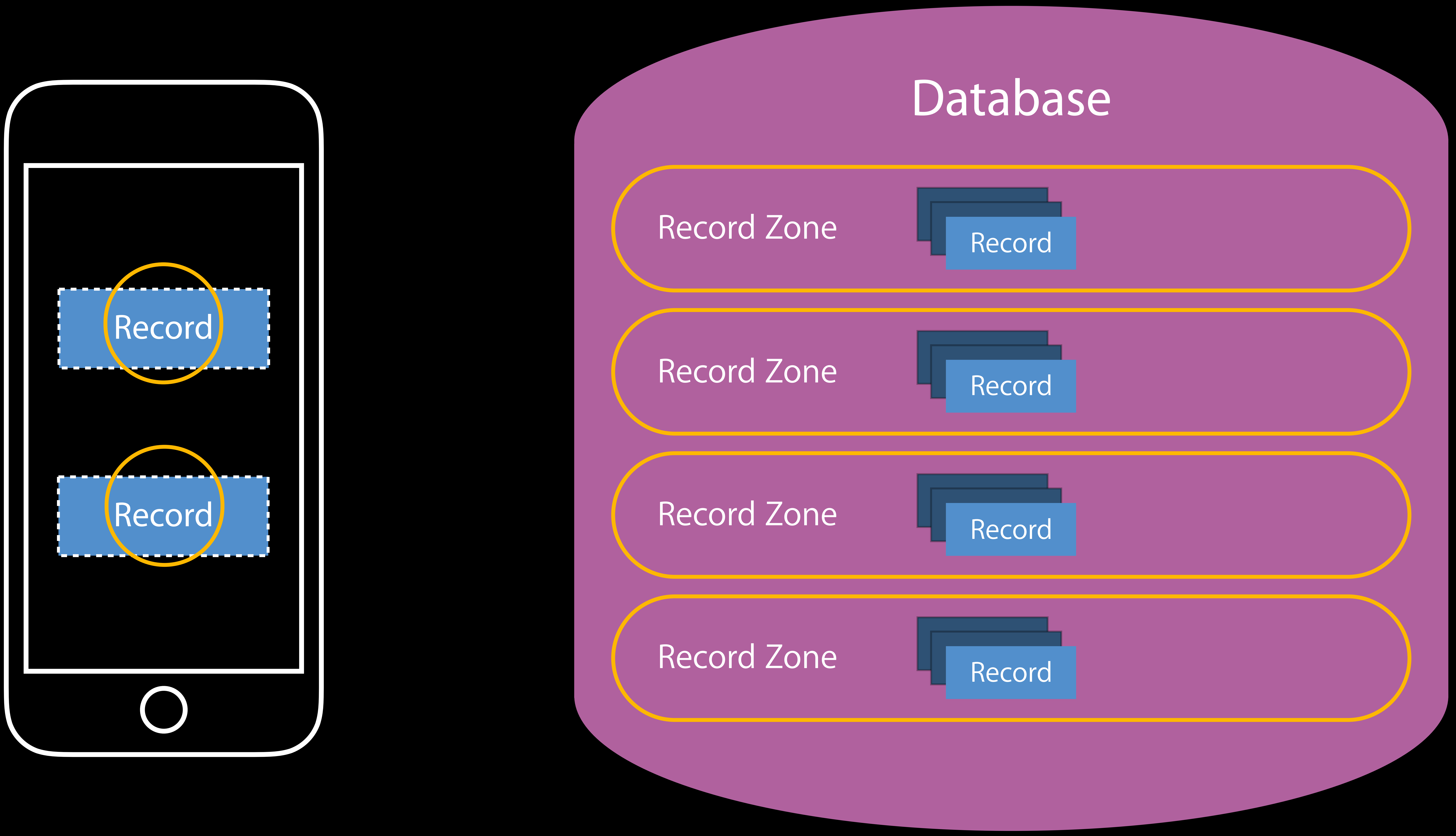
Handling Many Record Zones

CKFetchRecordZoneChangesOperation



Handling Many Record Zones

CKFetchRecordZoneChangesOperation



API Improvements

Long Lived Operations

CKOperation Timeouts

Handling Many Record Zones

Fetching Multiple Change Sets

API Improvements

Long Lived Operations

CKOperation Timeouts

Handling Many Record Zones

Fetching Multiple Change Sets

Fetch Multiple Change Sets

Remember this?

```
public class CKFetchRecordChangesOperation : CKDatabaseOperation {
    public var moreComing: Bool { get }
}
```

Fetch Multiple Change Sets

Remember this?

```
public class CKFetchRecordChangesOperation : CKDatabaseOperation {
   public var moreComing: Bool { get }
}
```

Client code responsible for fetching next batch

Fetch Multiple Change Sets

Remember this?

```
public class CKFetchRecordChangesOperation : CKDatabaseOperation {
    public var moreComing: Bool { get }
}
```

Client code responsible for fetching next batch

CloudKit idle

Fetch Multiple Change Sets

Use this instead

```
public class CKFetchRecordZoneChangesOperation : CKDatabaseOperation {


   public var fetchAllChanges: Bool
}
```

Fetch Multiple Change Sets

Use this instead

```
public class CKFetchRecordZoneChangesOperation : CKDatabaseOperation {


   public var fetchAllChanges: Bool
}
```

CloudKit keeps pipeline full

Fetch Multiple Change Sets

Use this instead

```
public class CKFetchRecordZoneChangesOperation : CKDatabaseOperation {
   // fetchAllChanges is true by default
   public var fetchAllChanges: Bool
}
```

CloudKit keeps pipeline full

Fetch Multiple Change Sets

New callback

```
public class CKFetchRecordZoneChangesOperation : CKDatabaseOperation {
   public var recordZoneChangeTokensUpdatedBlock: ((CKRecordZoneID, CKServerChangeToken?,
      Data?) -> Void)?
}
```

Fetch Multiple Change Sets

New callback

```
public class CKFetchRecordZoneChangesOperation : CKDatabaseOperation {
   public var recordZoneChangeTokensUpdatedBlock: ((CKRecordZoneID, CKServerChangeToken?,
      Data?) -> Void)?
}
```

Earlier record changes are safe to commit

Fetch Multiple Change Sets

New callback

```
public class CKFetchRecordZoneChangesOperation : CKDatabaseOperation {
   public var recordZoneChangeTokensUpdatedBlock: ((CKRecordZoneID, CKServerChangeToken?,
      Data?) -> Void)?
}
```

Earlier record changes are safe to commit

New server change token can be used on a new CKFetchRecordZoneChangesOperation

API Improvements

Long Lived Operations

CKOperation Timeouts

Handling Many Record Zones

Fetching Multiple Change Sets

Sharing UI

You and I are going to share some records

Jacob Farkas

Sharing Records

```
public class CKShare : CKRecord
```

Sharing Records

What is shared?

```
public class CKShare : CKRecord
```

Sharing Records

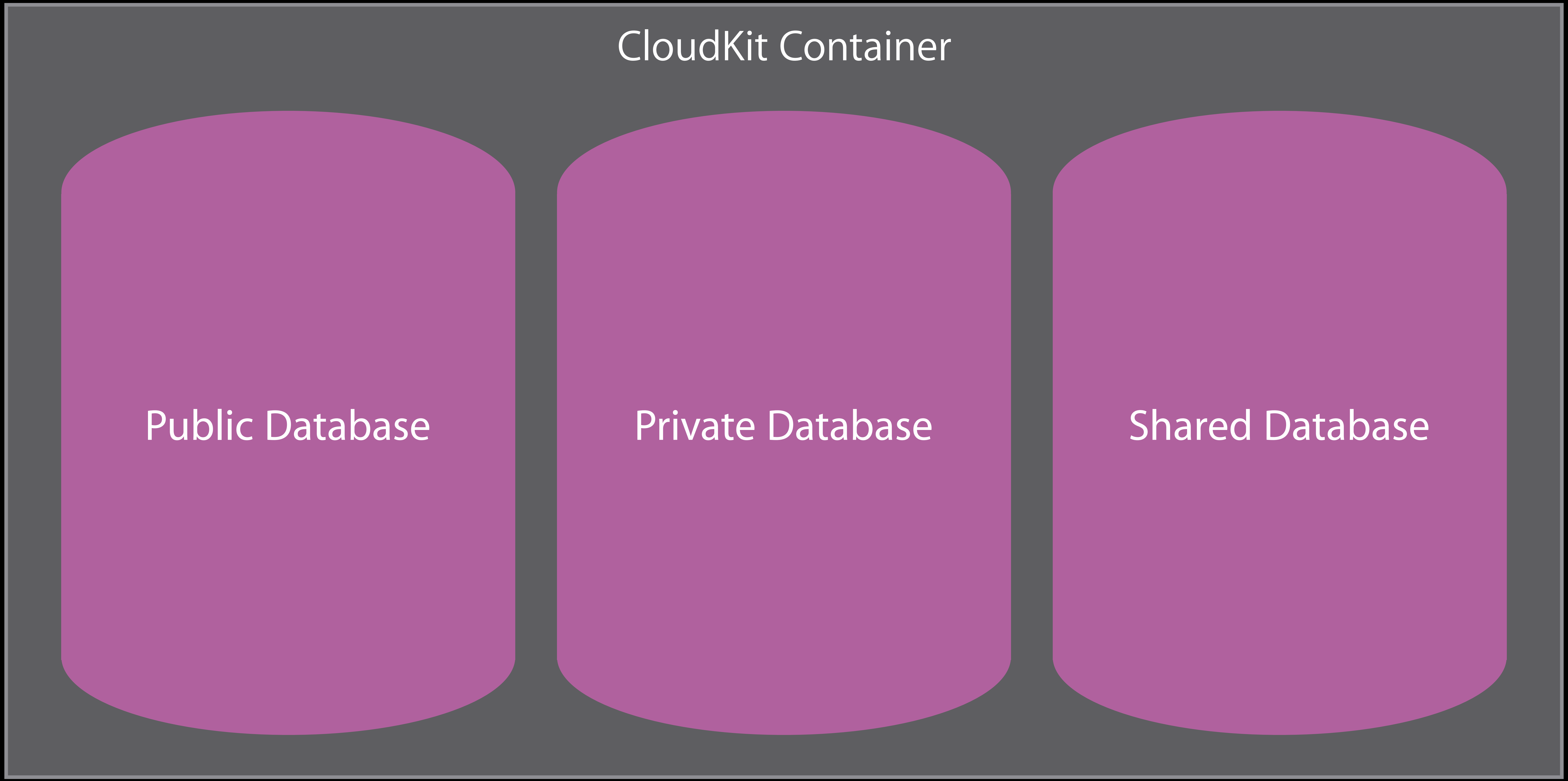
What is shared?

Who is it shared with?

```
public class CKShare : CKRecord
```

Sharing Overview

Shared database

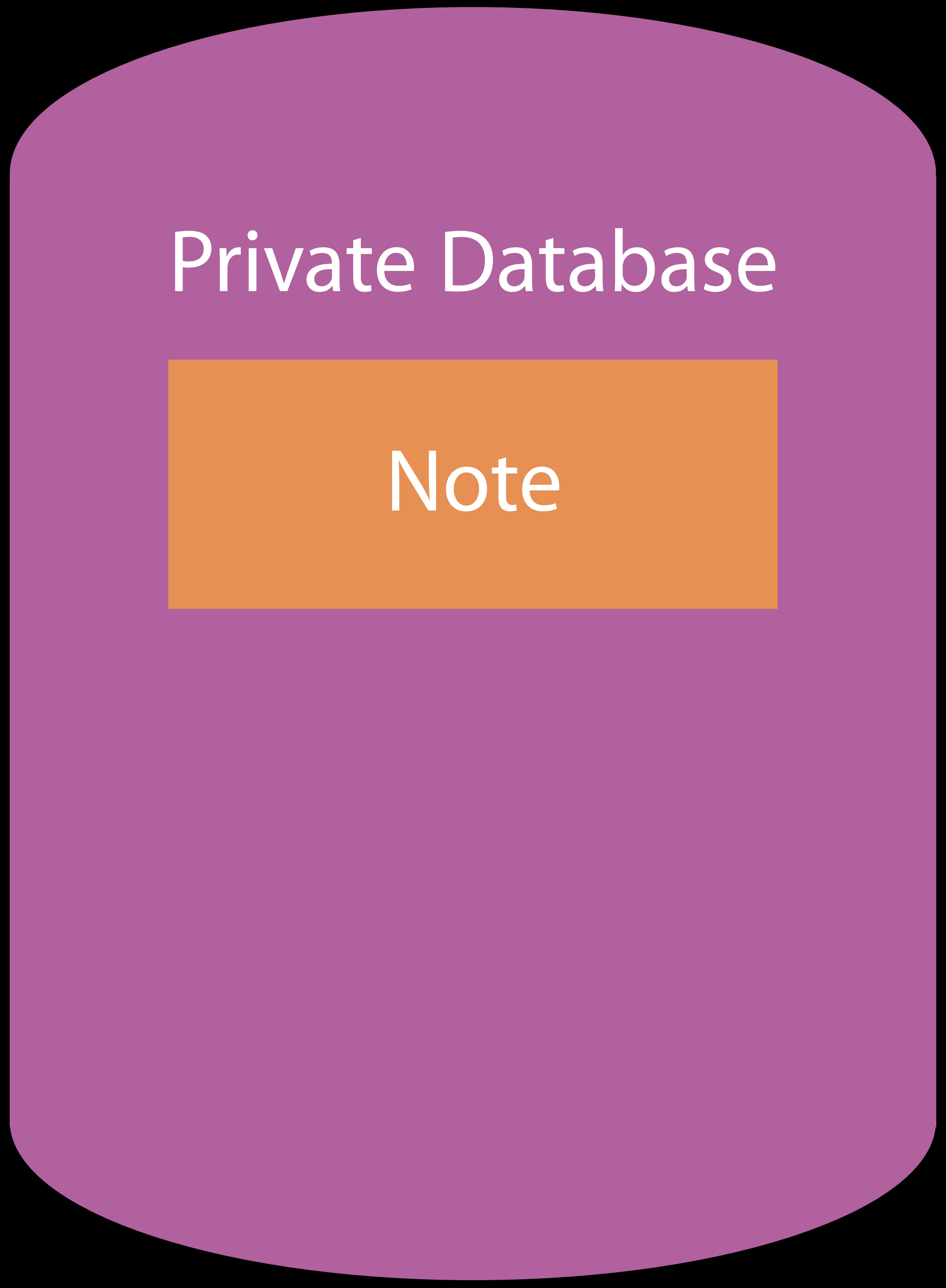


Sharing Records

What is shared?

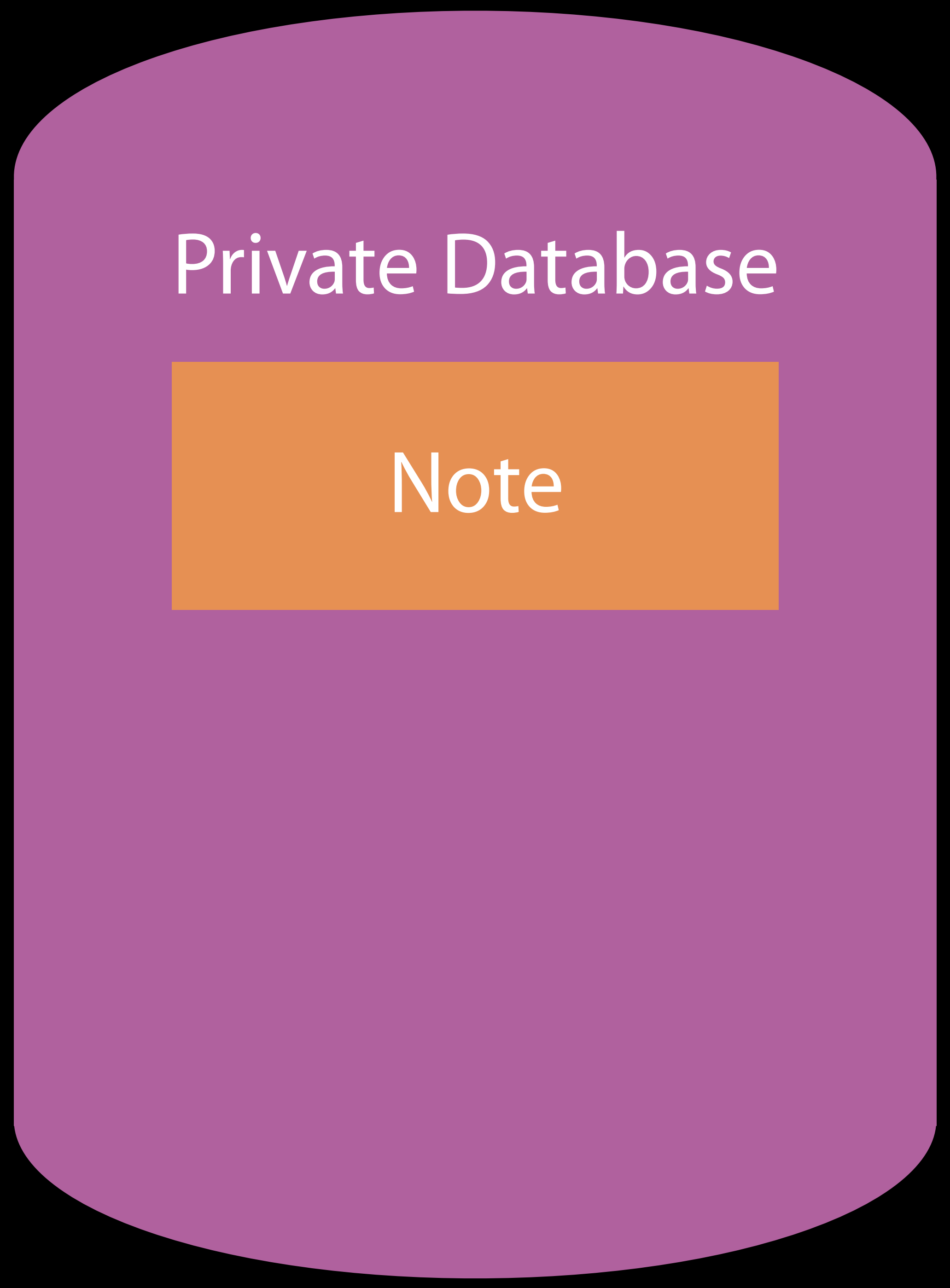
Sharing Records

What is shared?



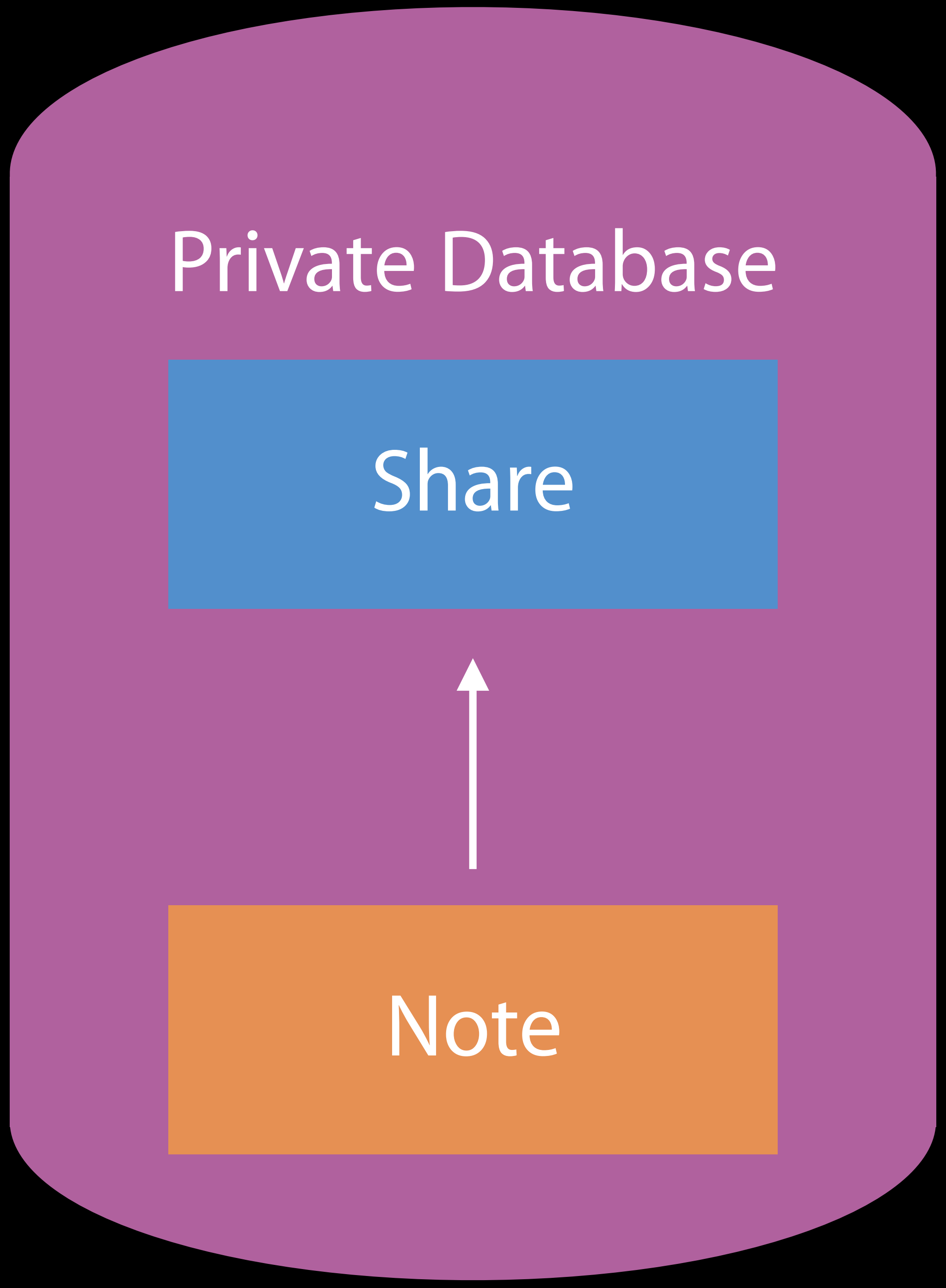
Sharing Records

What is shared?



Sharing Records

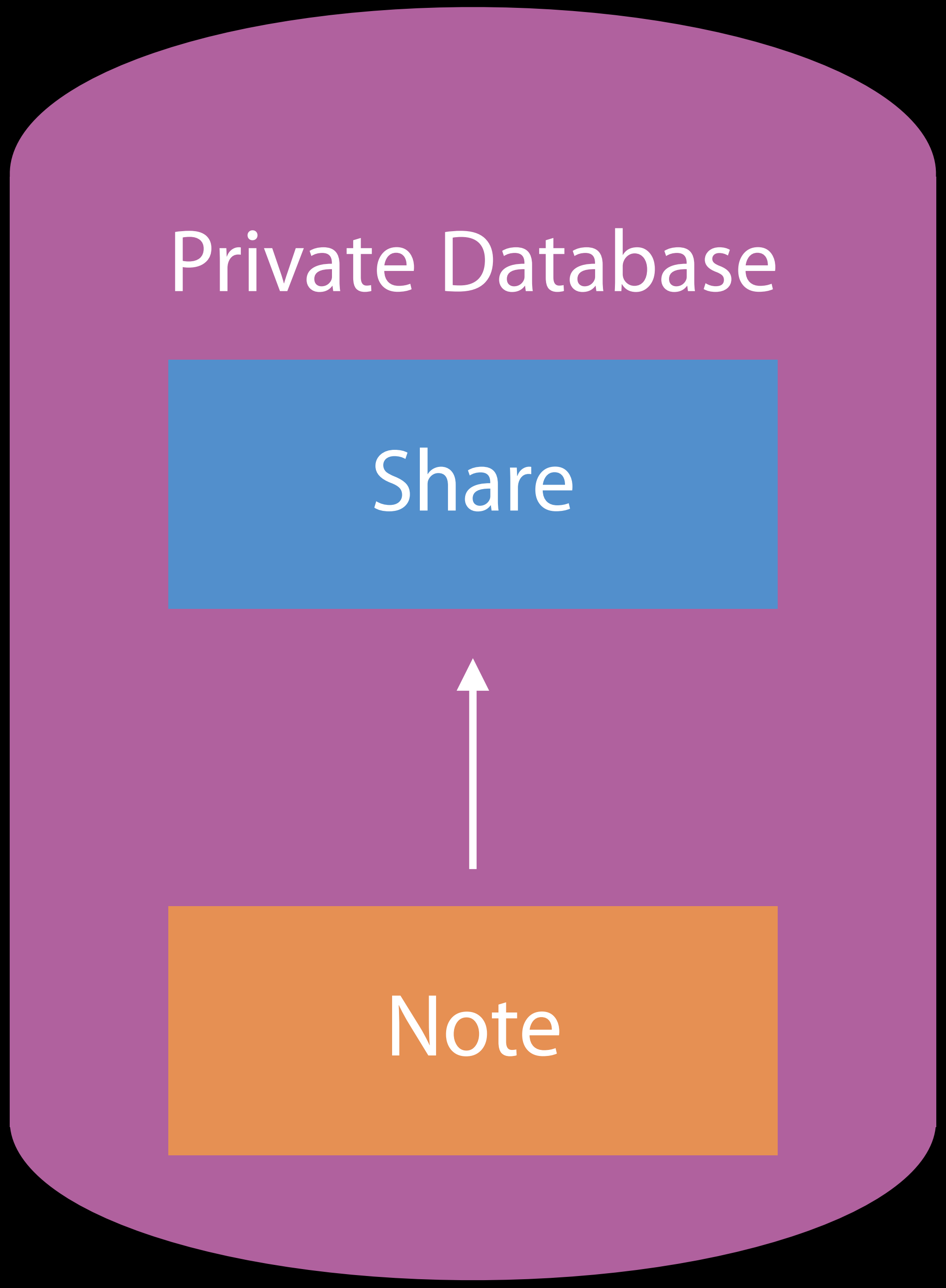
What is shared?



```
public class CKShare : CKRecord {
    public init(rootRecord : CKRecord)
}
```

Sharing Records

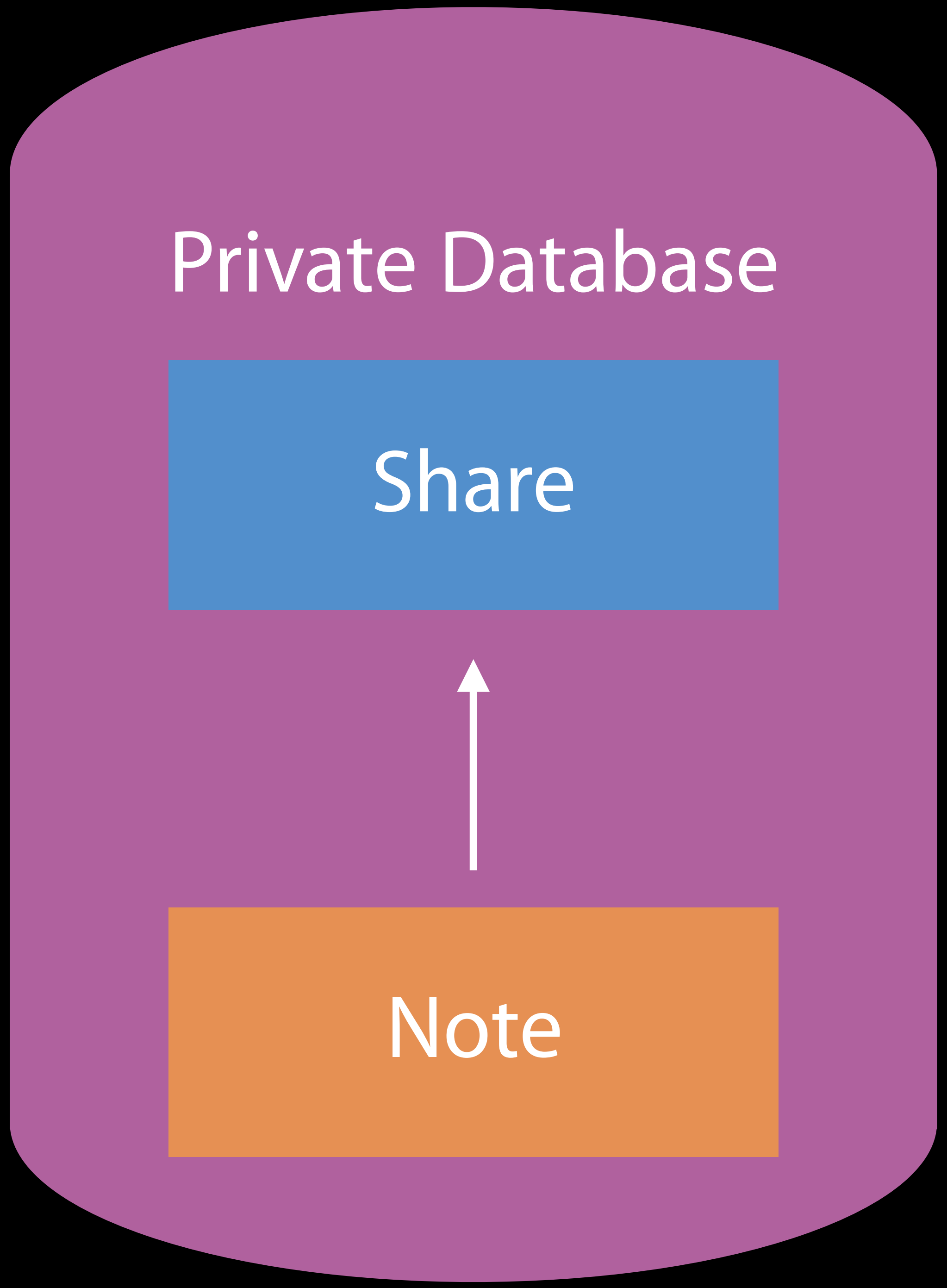
What is shared?



```
public class CKShare : CKRecord {
    public init(rootRecord : CKRecord)
}
```

Sharing Records

What is shared?

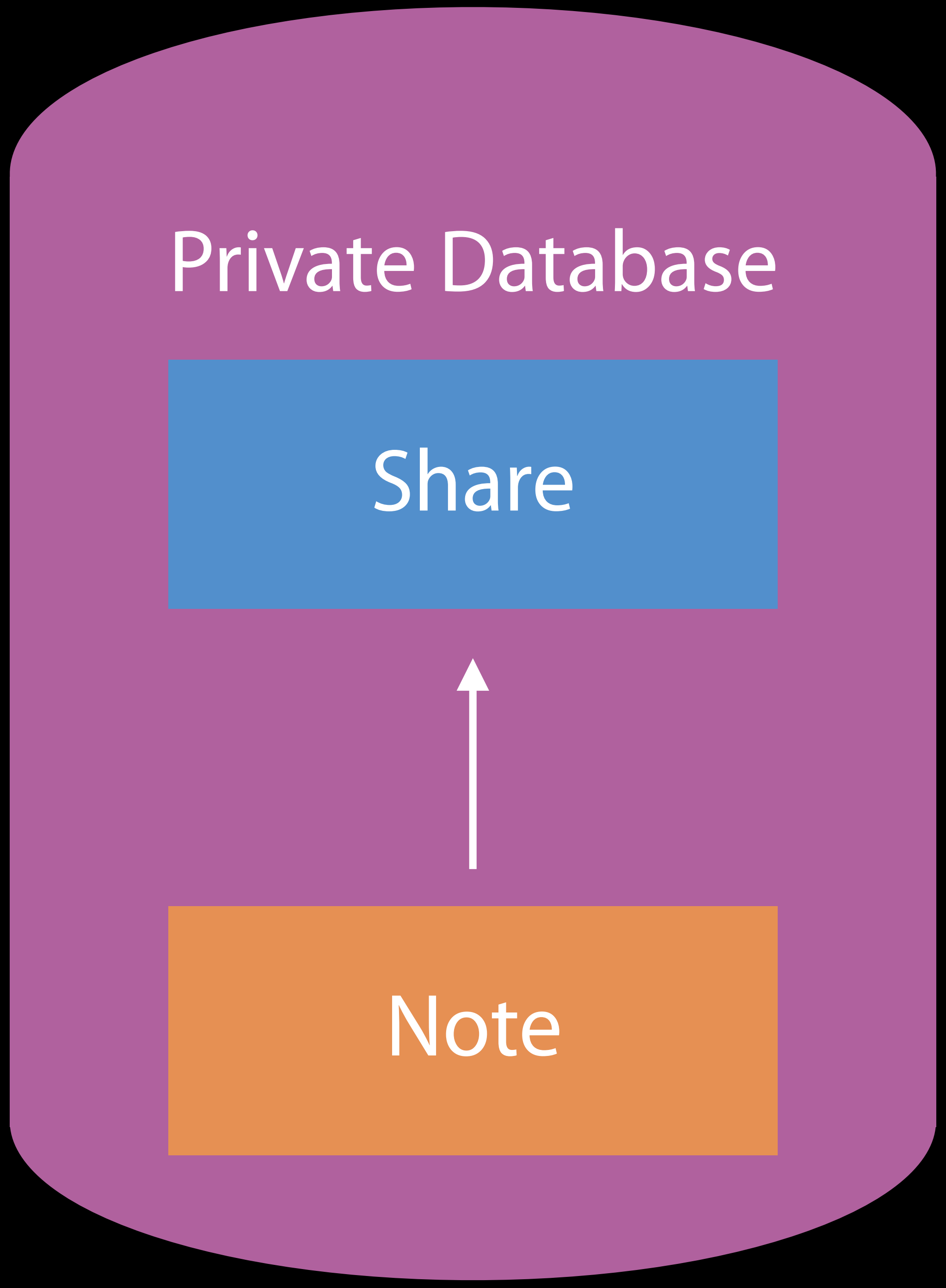


```
public class CKShare : CKRecord {
   public init(rootRecord : CKRecord)
}
```

```
public class CKRecord {
   public var share: CKReference? { get }
}
```

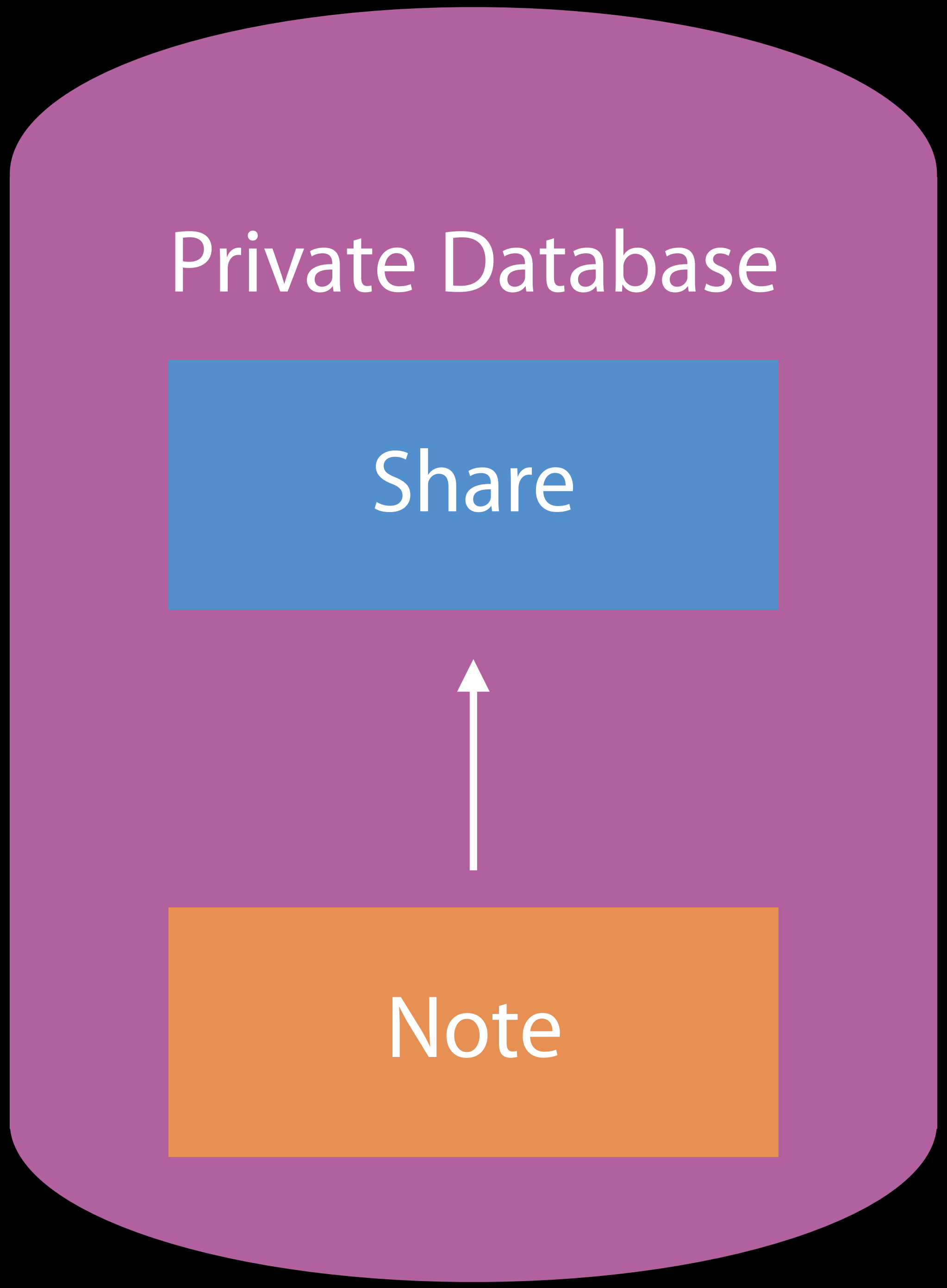
Sharing Records

Who is it shared with?



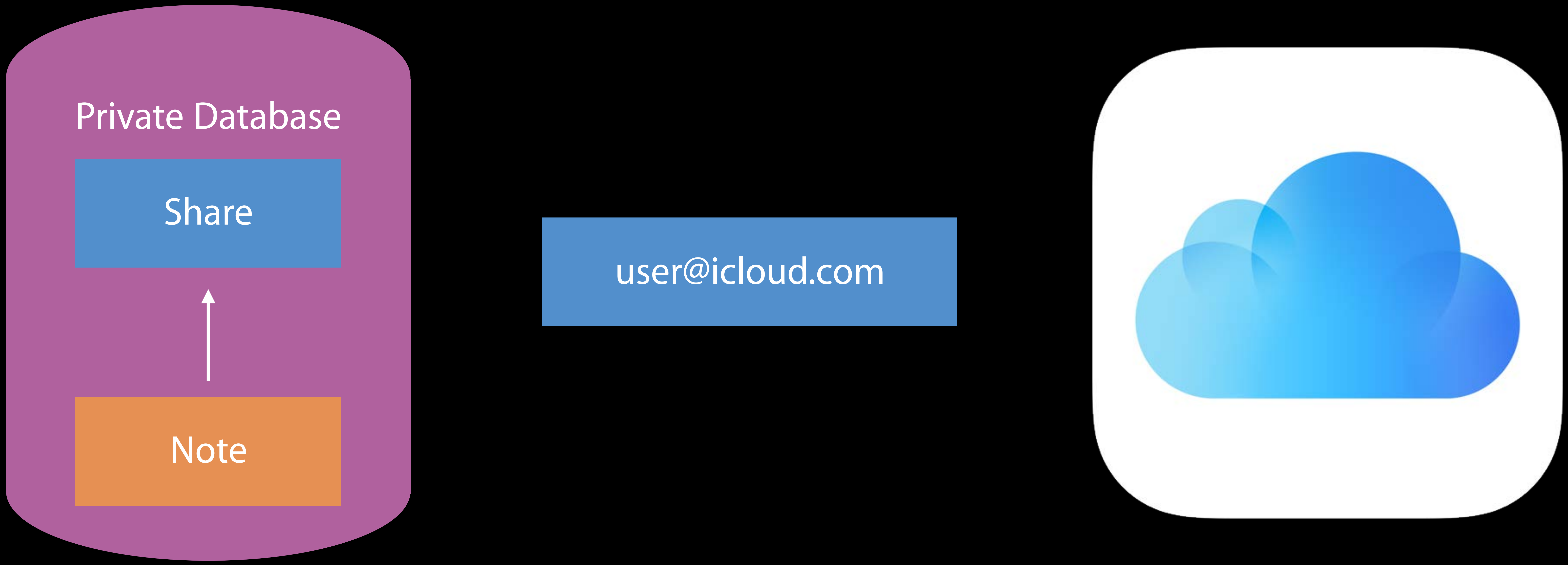
Sharing Records

Who is it shared with?



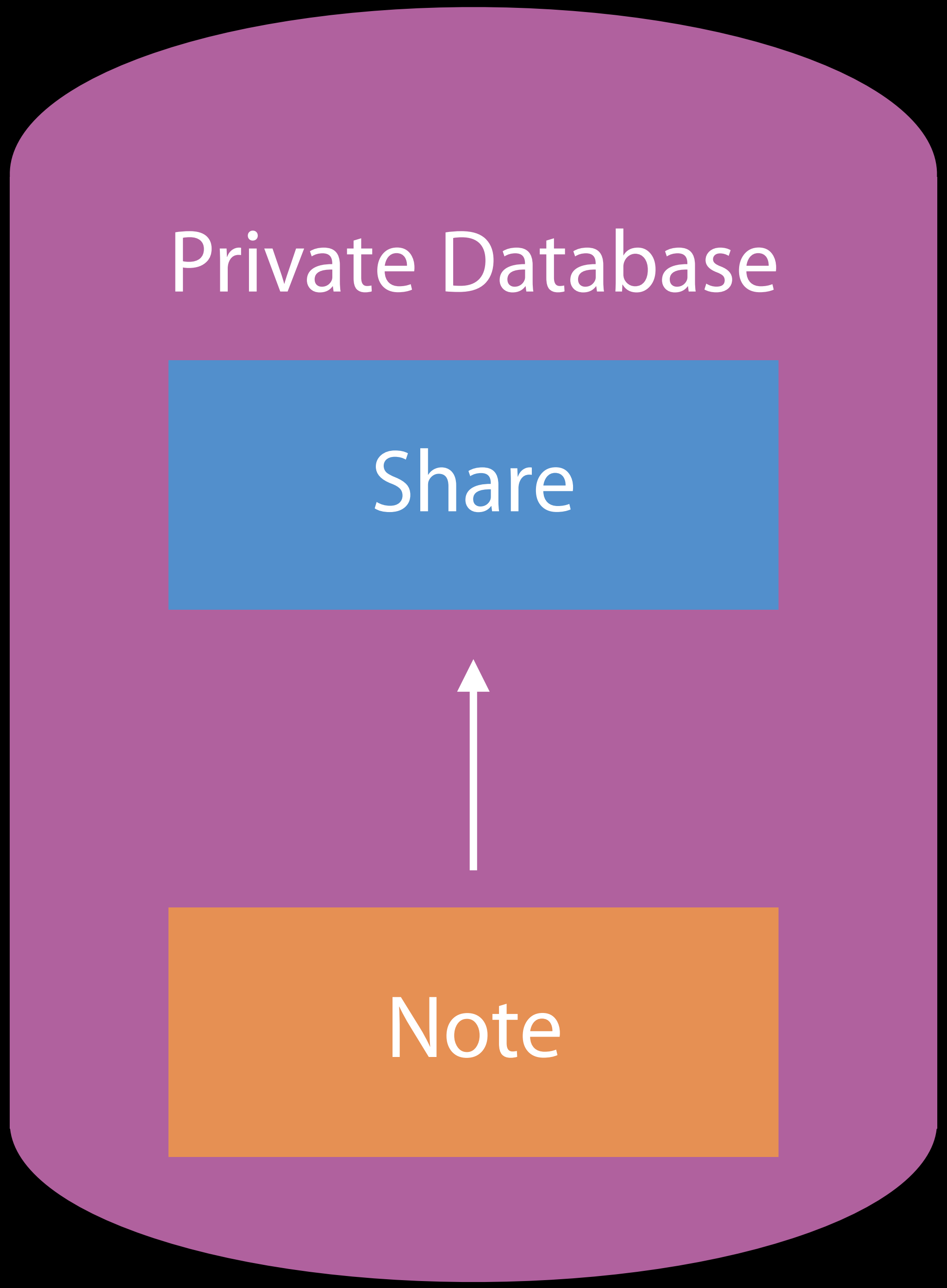
Sharing Records

Who is it shared with?



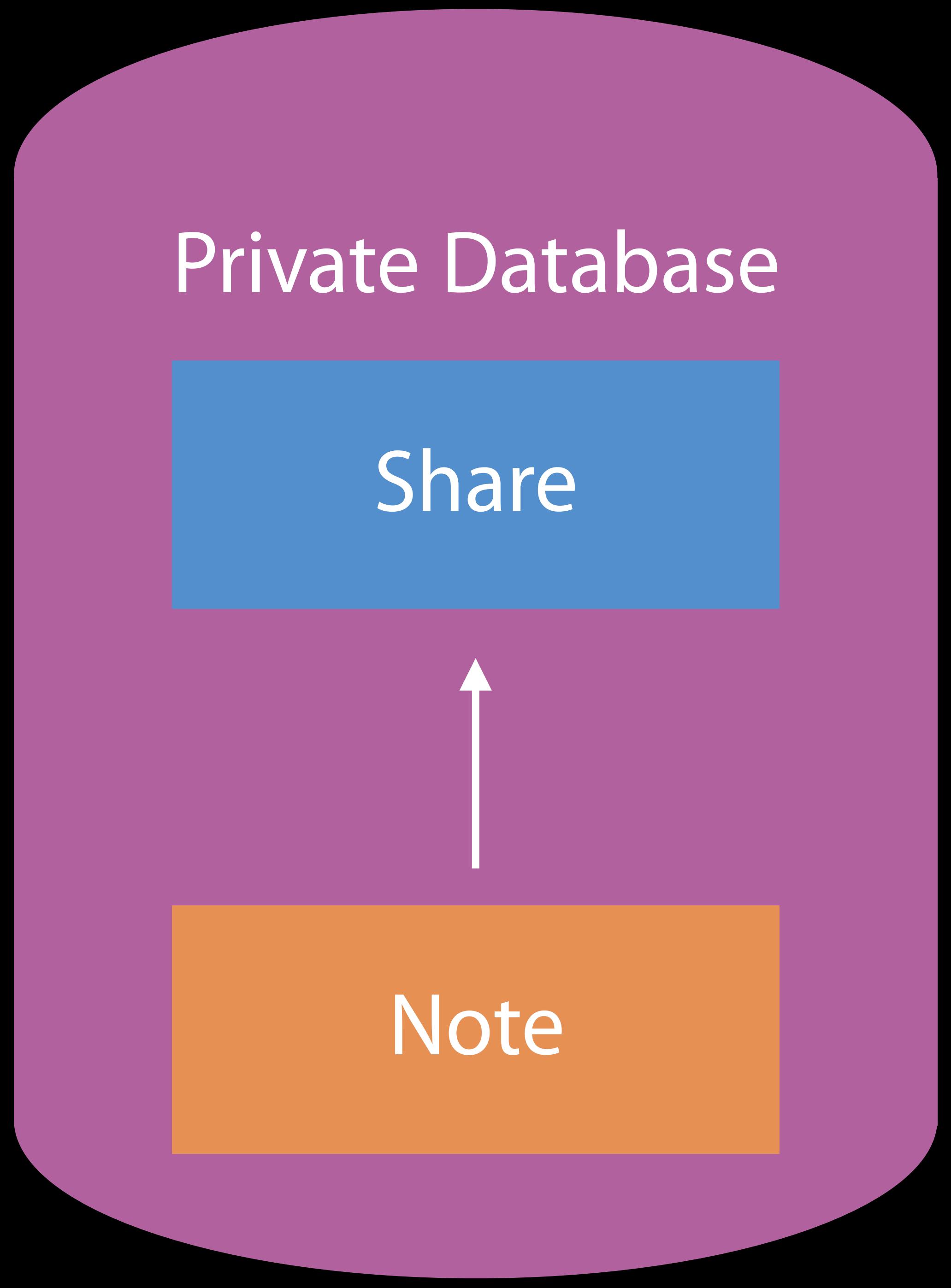
Sharing Records

Who is it shared with?



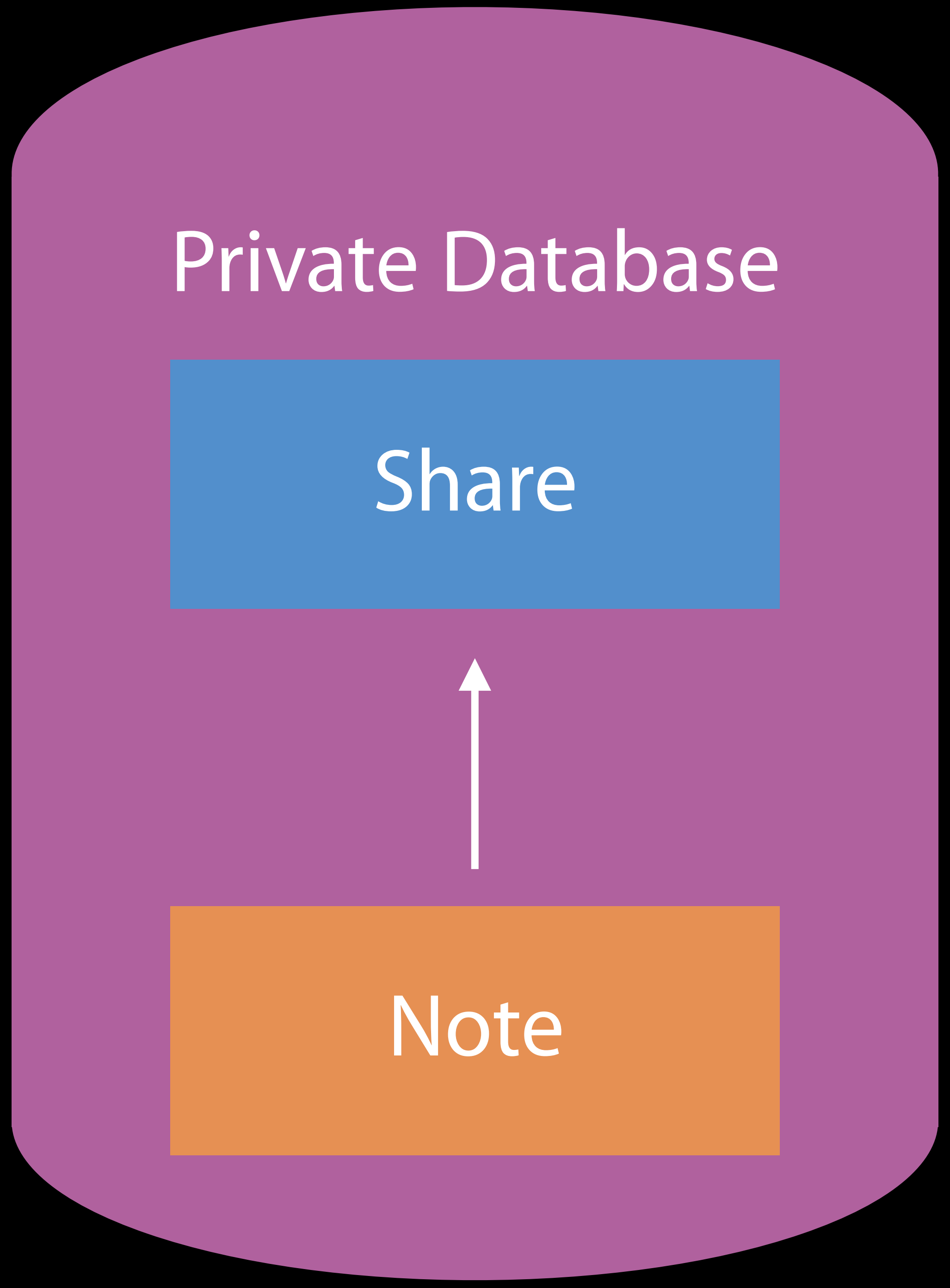
Sharing Records

Who is it shared with?



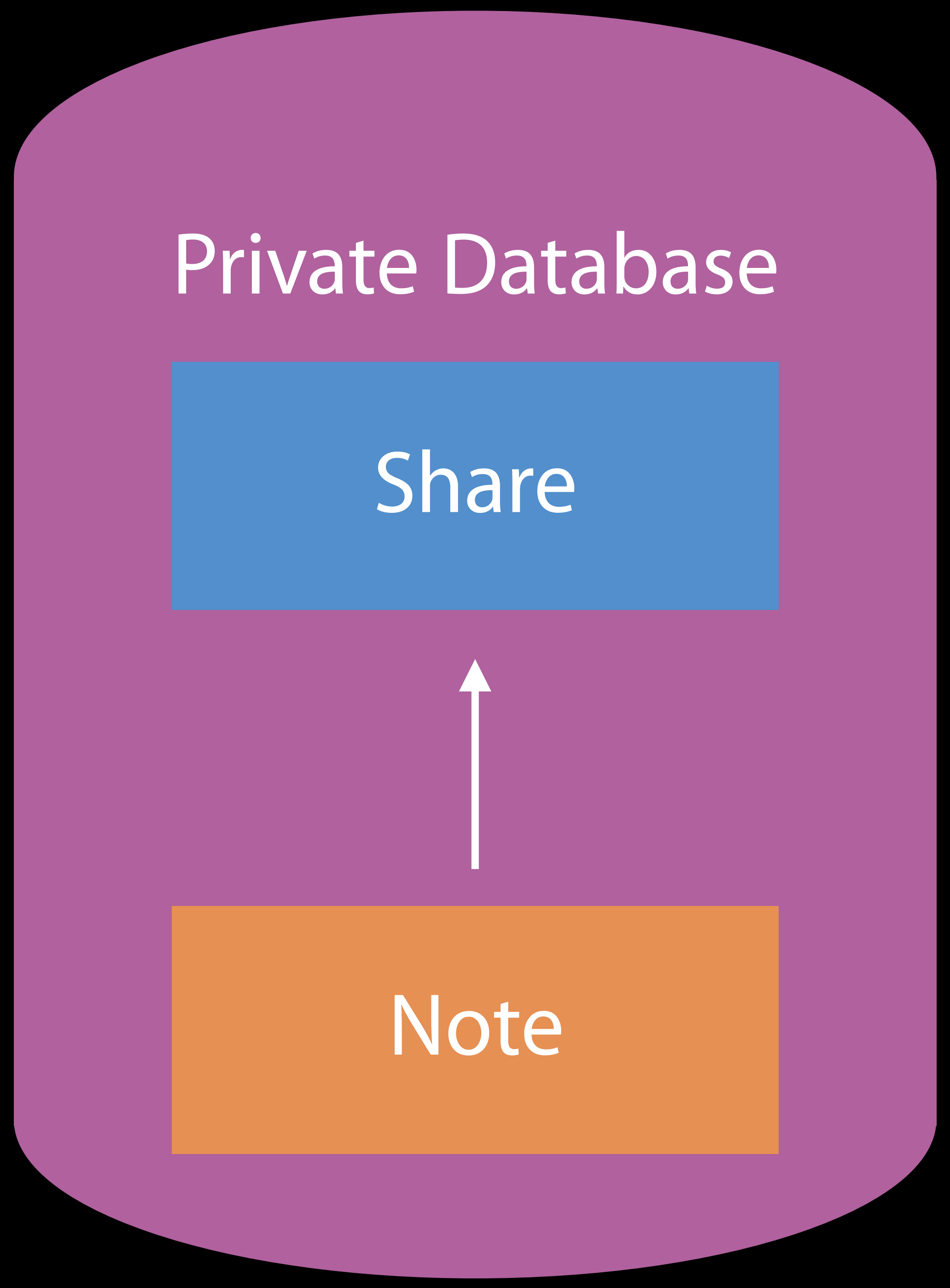
Sharing Records

Who is it shared with?



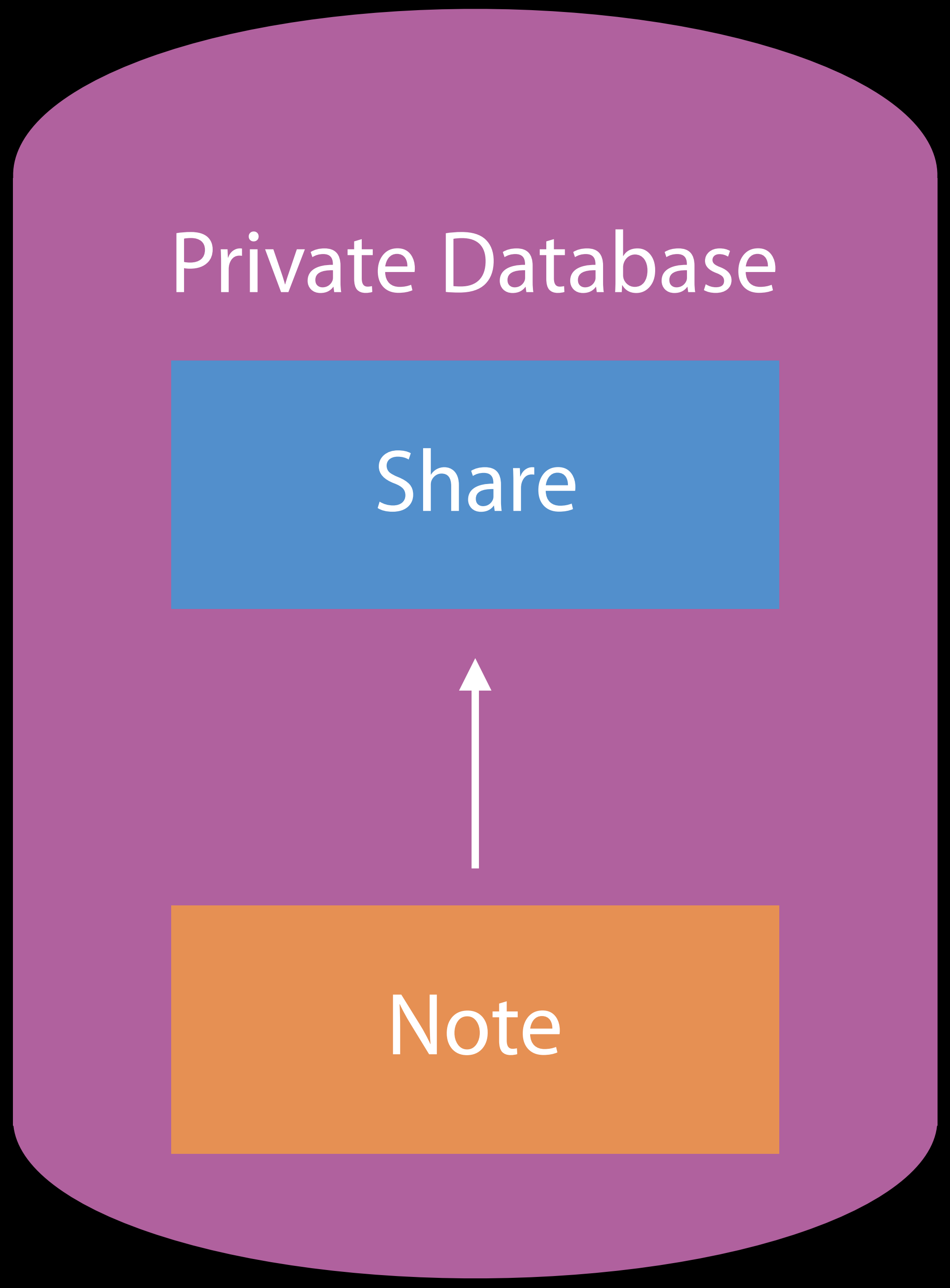
Sharing Records

Who is it shared with?



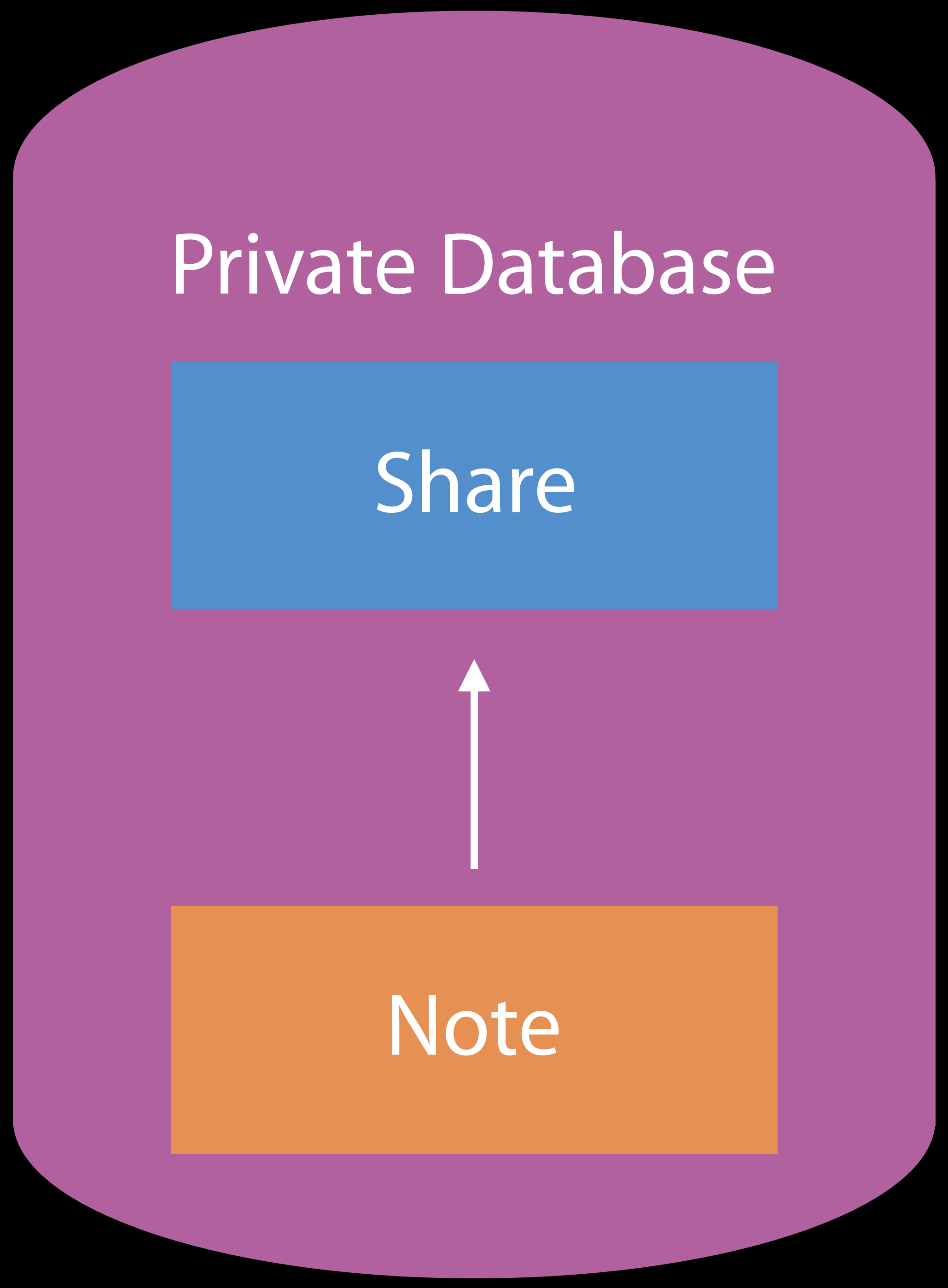
Sharing Records

Share URLs



Sharing Records

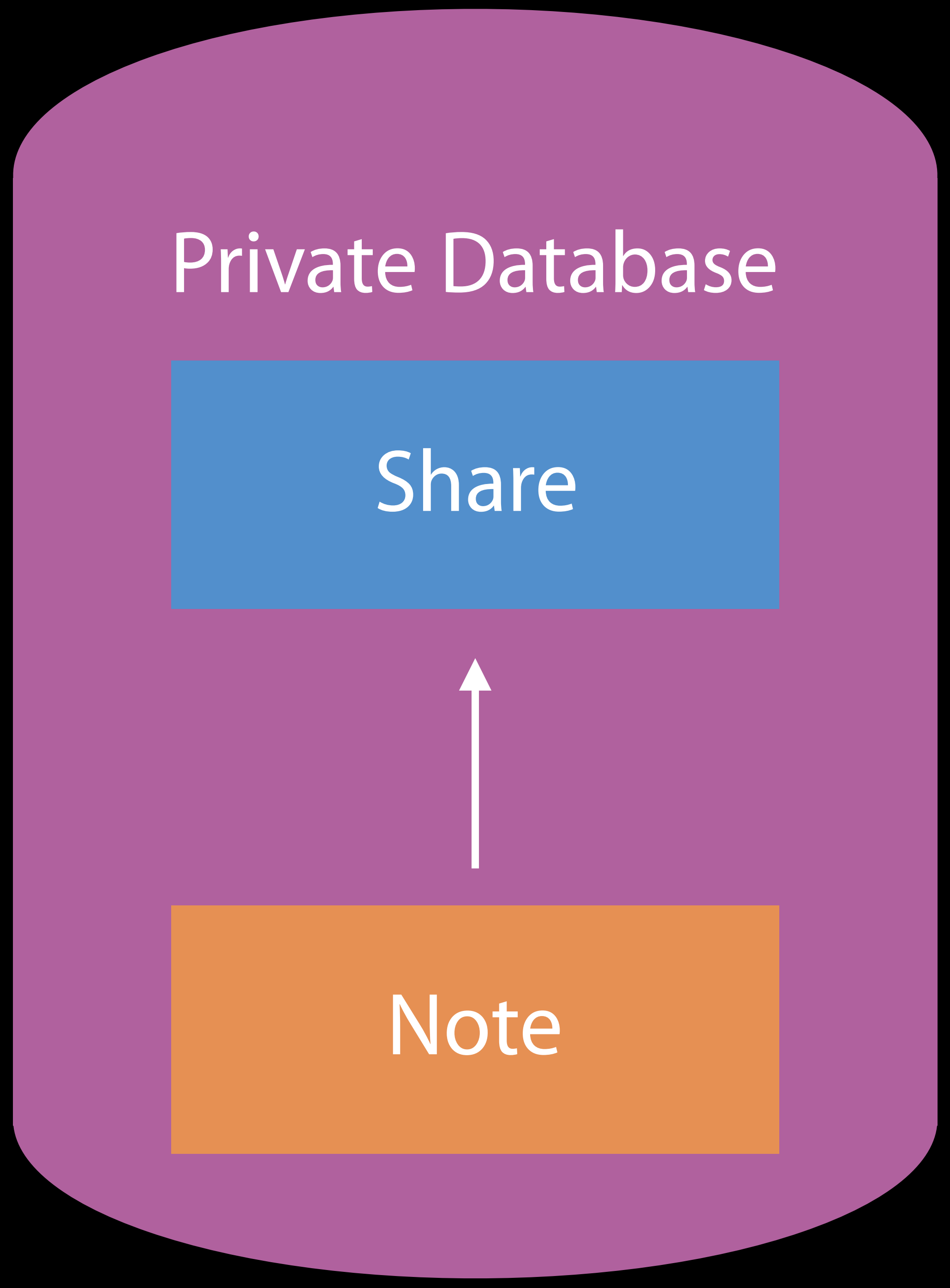
Share URLs

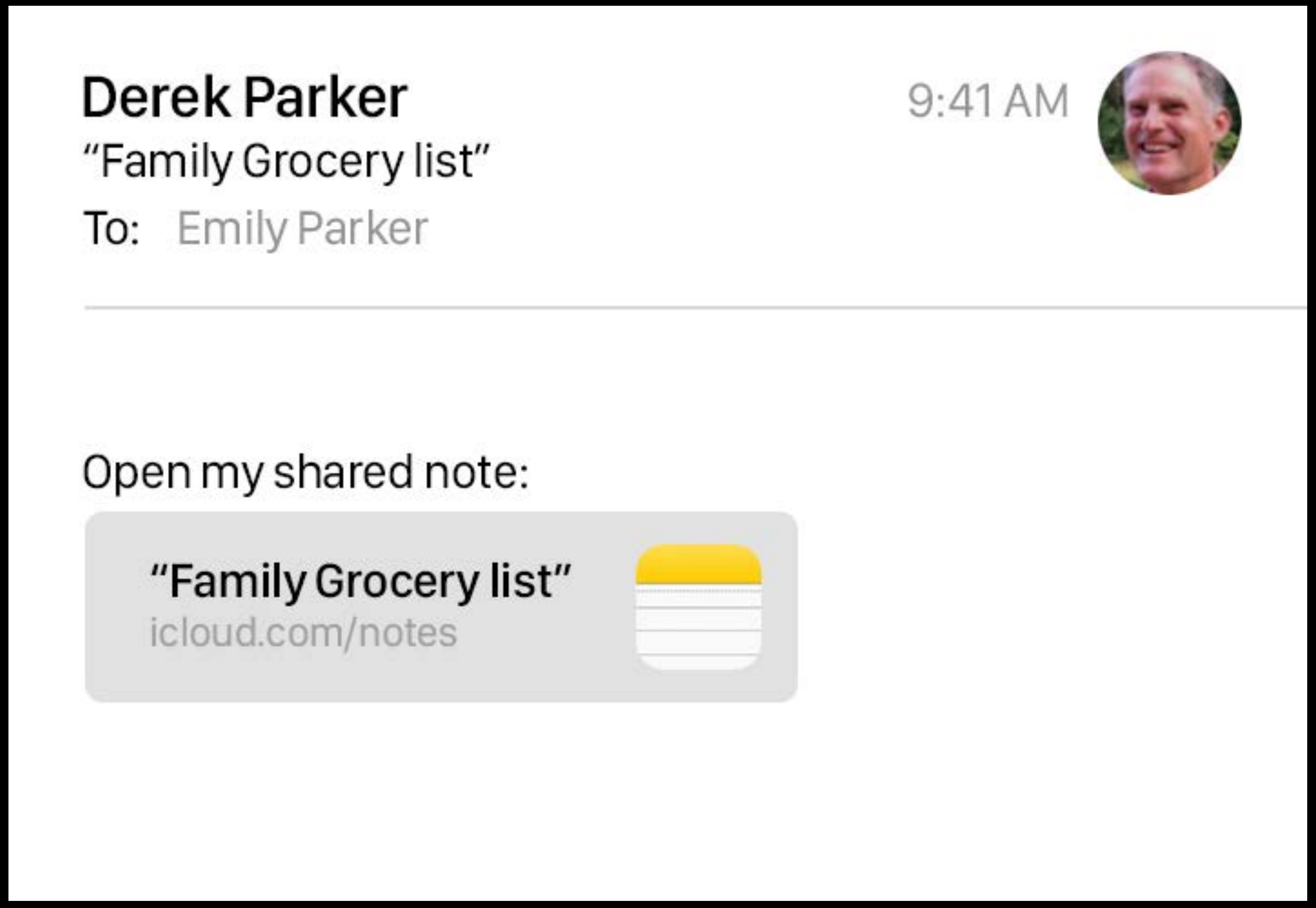


```
https://www.icloud.com/notes/000Y4qow0owP6NOxDzs4qgi8Q
```

Sharing Records

Share URLs




Derek Parker
"Family Grocery list"
To: Emily Parker

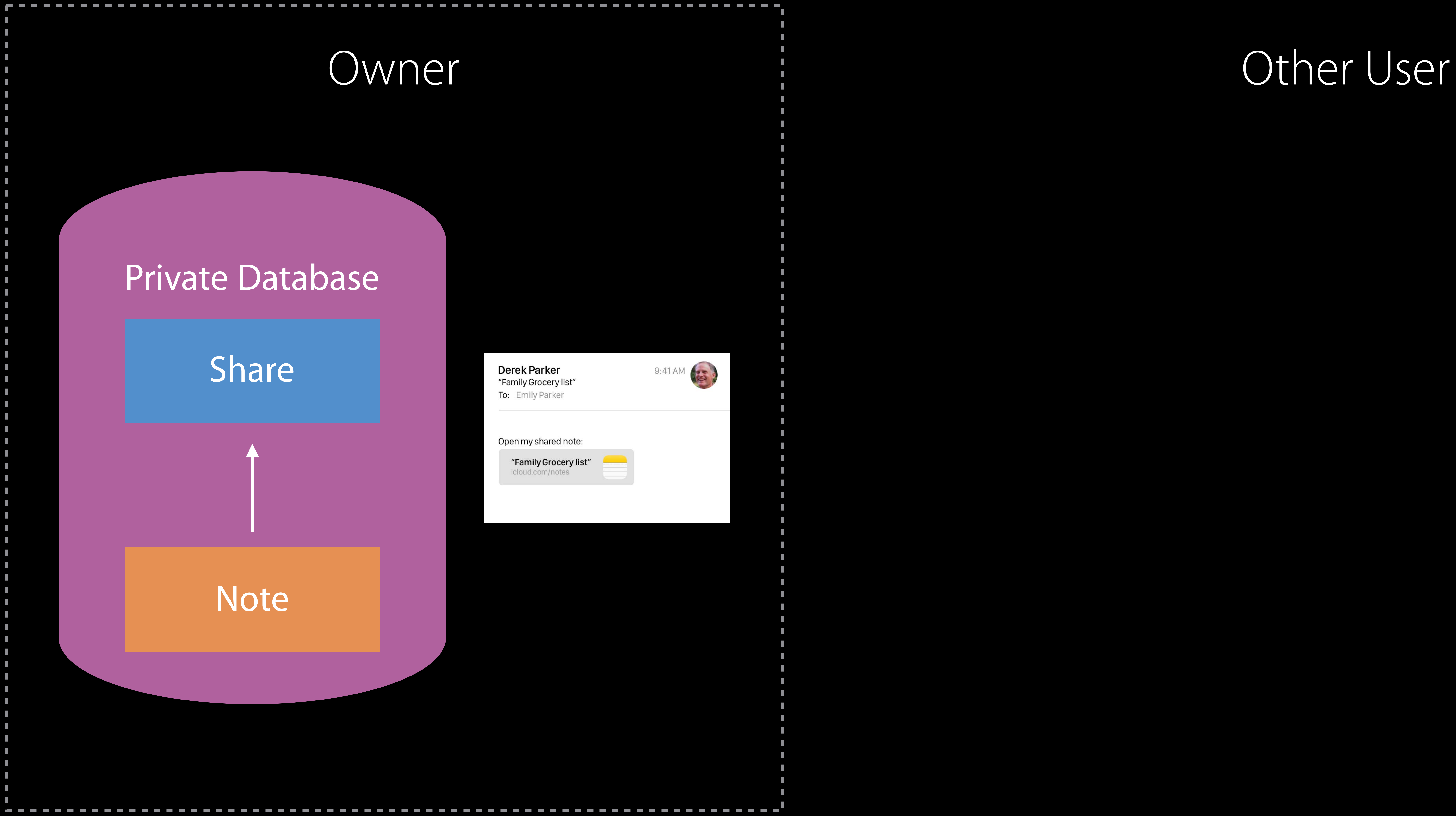
9:41 AM

Open my shared note:

"Family Grocery list"
icloud.com/notes

Sharing Records

Accepting a share

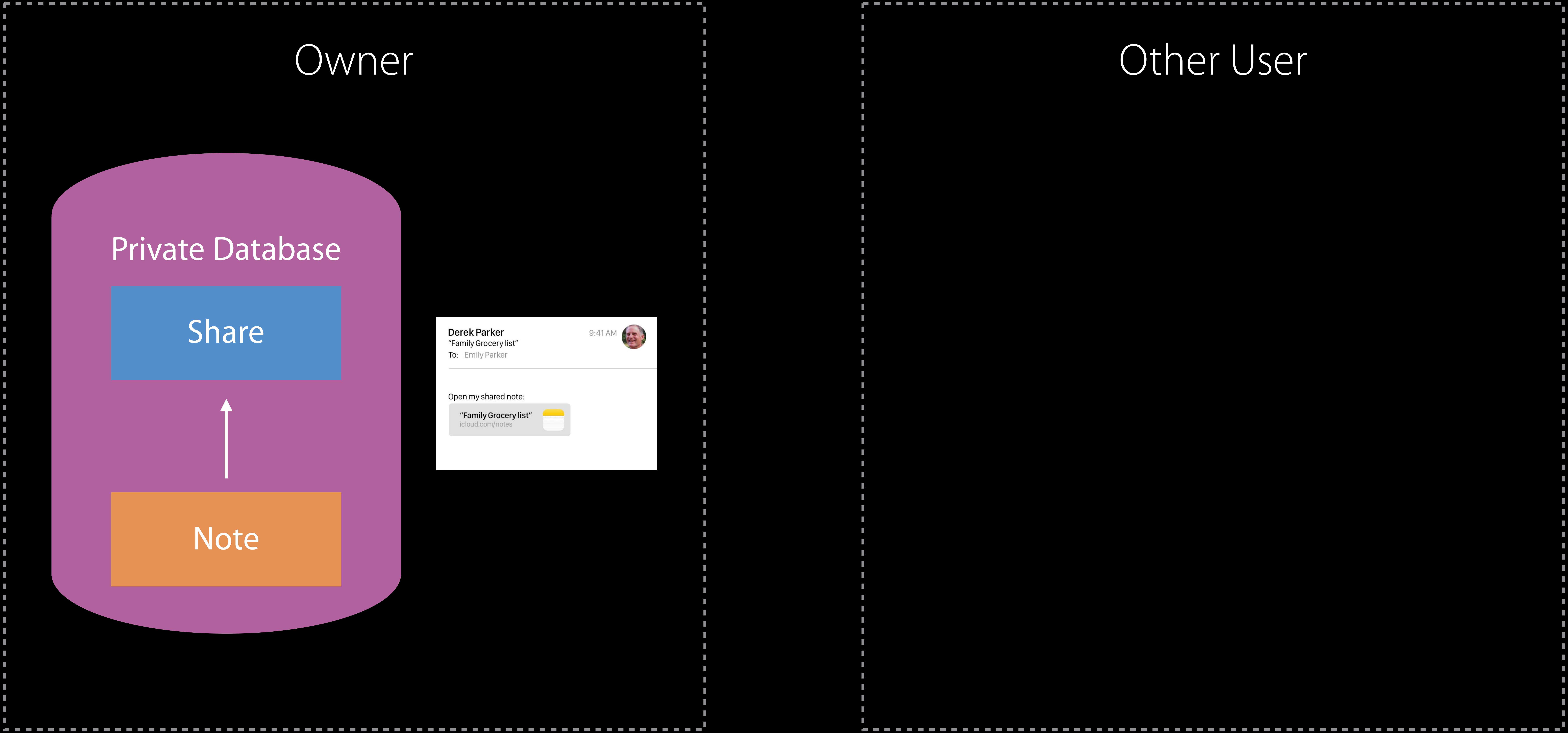


Derek Parker 9:41 AM
"Family Grocery list"
To: Emily Parker

Open my shared note:
"Family Grocery list"
icloud.com/notes

Sharing Records

Accepting a share

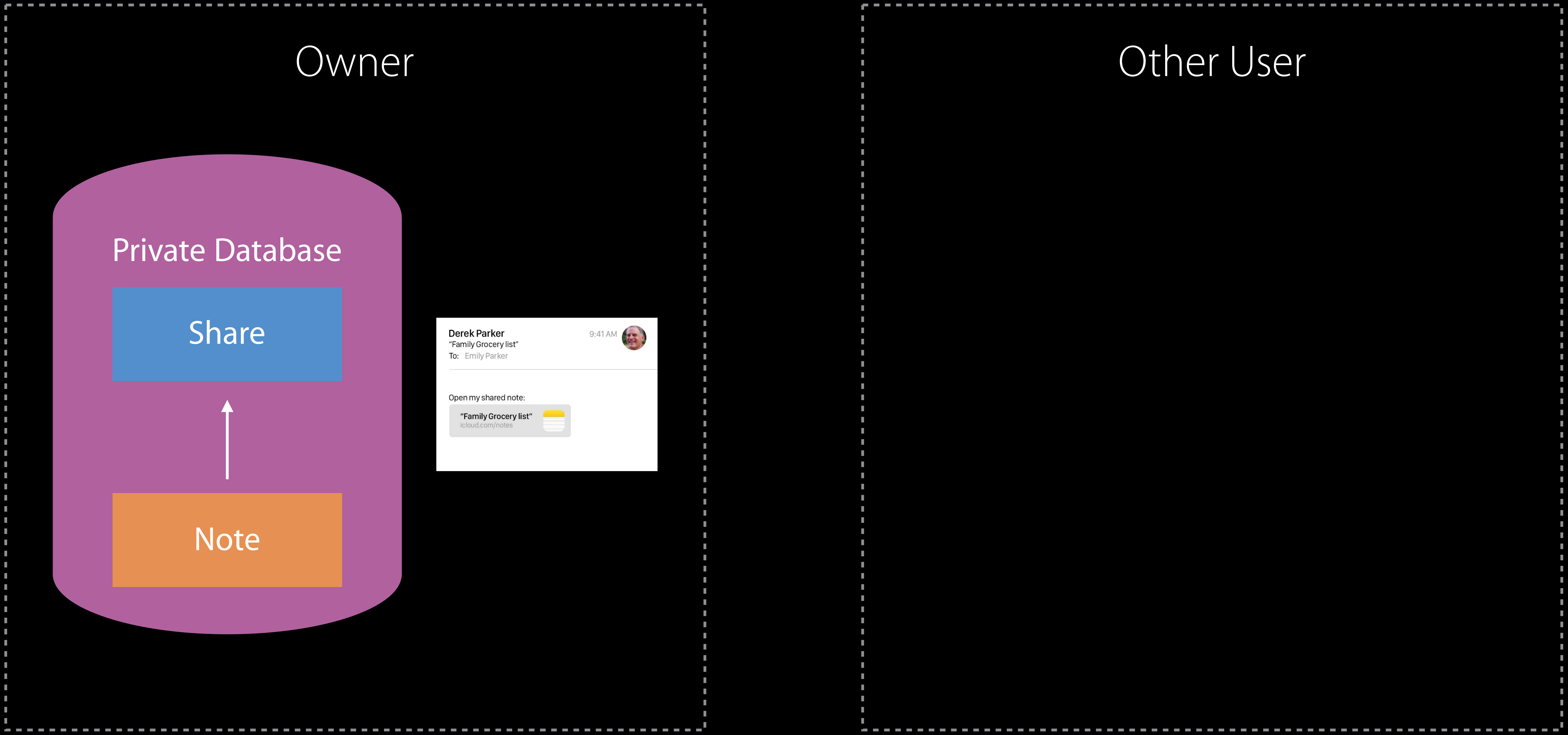


Derek Parker 9:41 AM
"Family Grocery list"
To: Emily Parker

Open my shared note:
"Family Grocery list"
icloud.com/notes

Sharing Records

Accepting a share

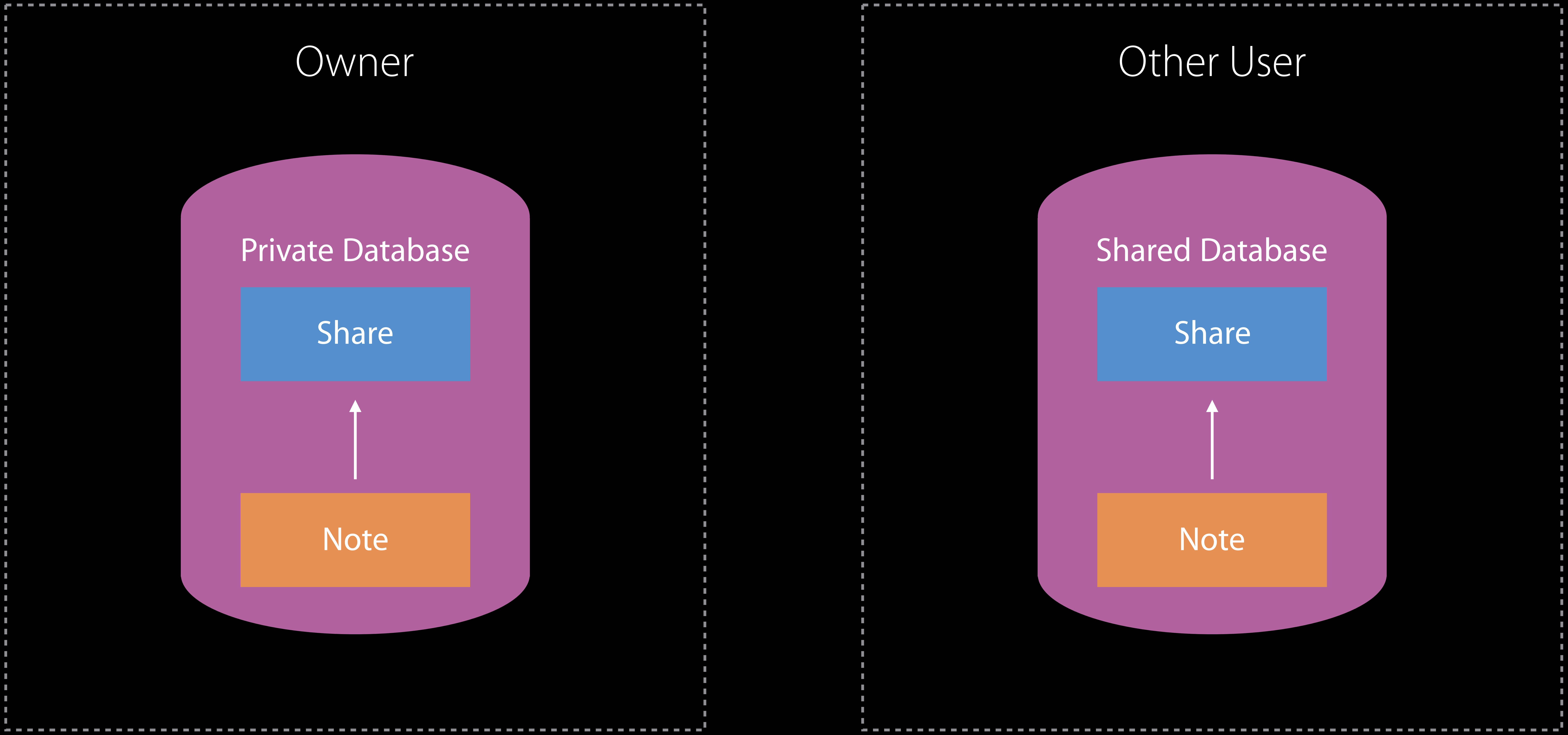


Derek Parker 9:41 AM
"Family Grocery list"
To: Emily Parker

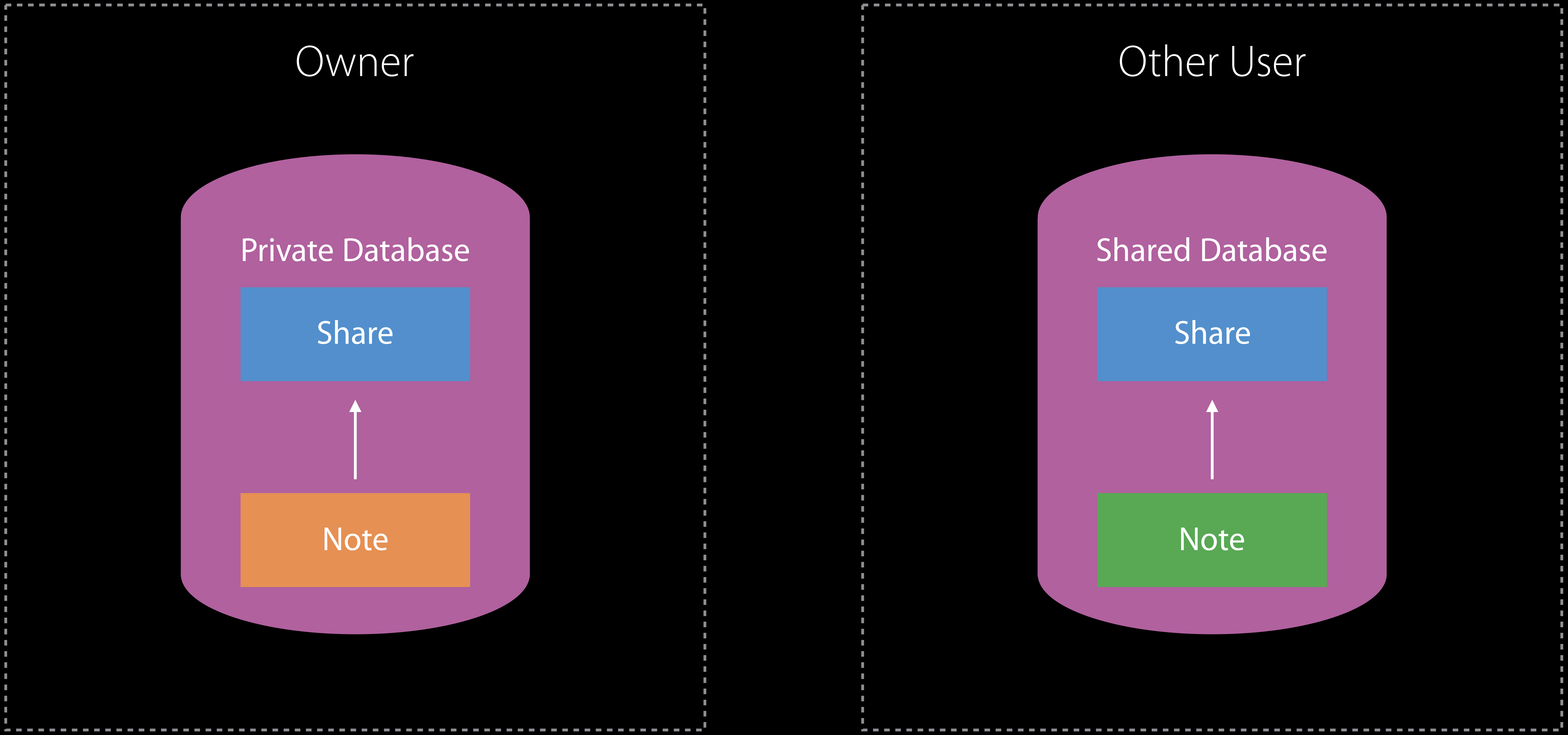
Open my shared note:
"Family Grocery list"
icloud.com/notes

Sharing Records

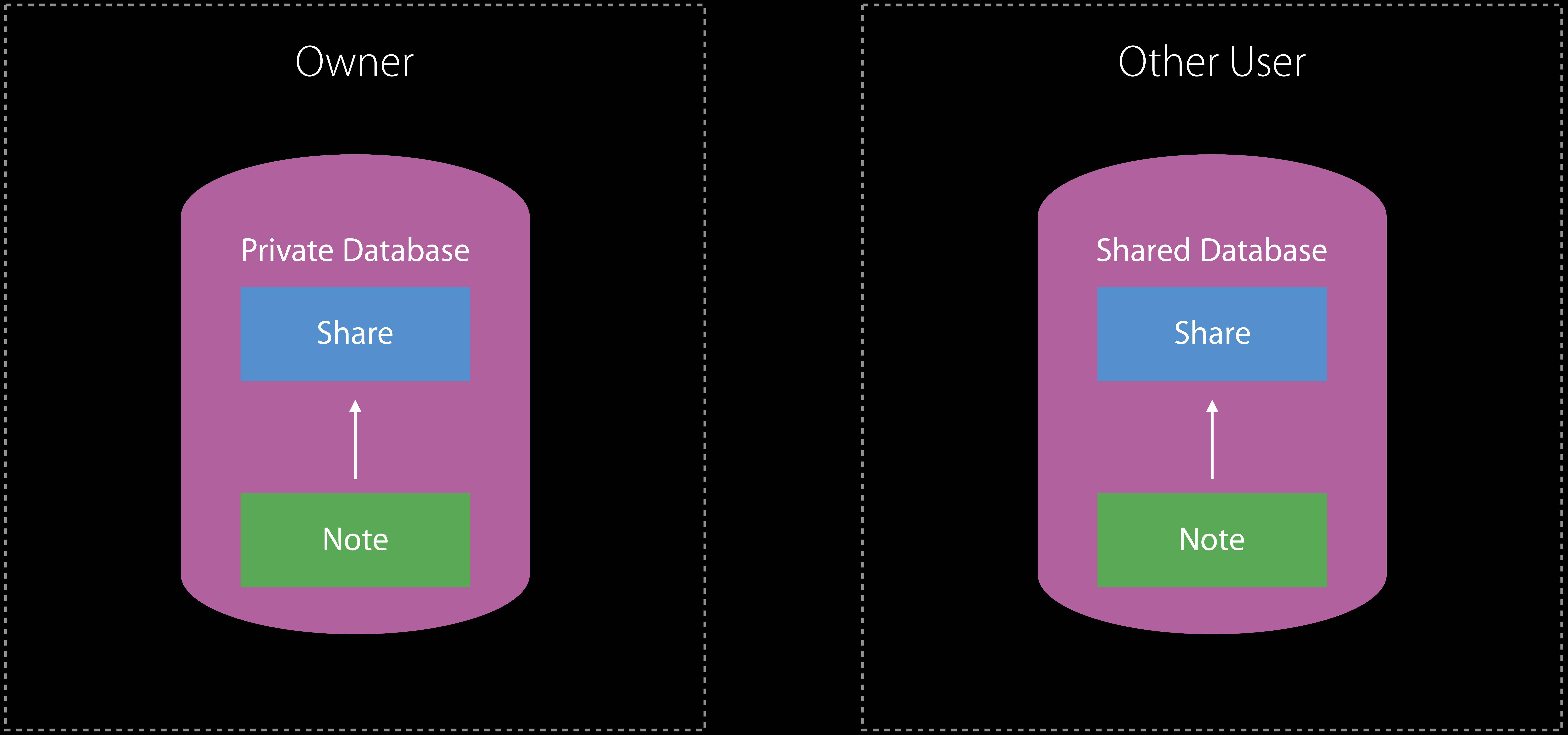
Accepting a share



Sharing Records



Sharing Records



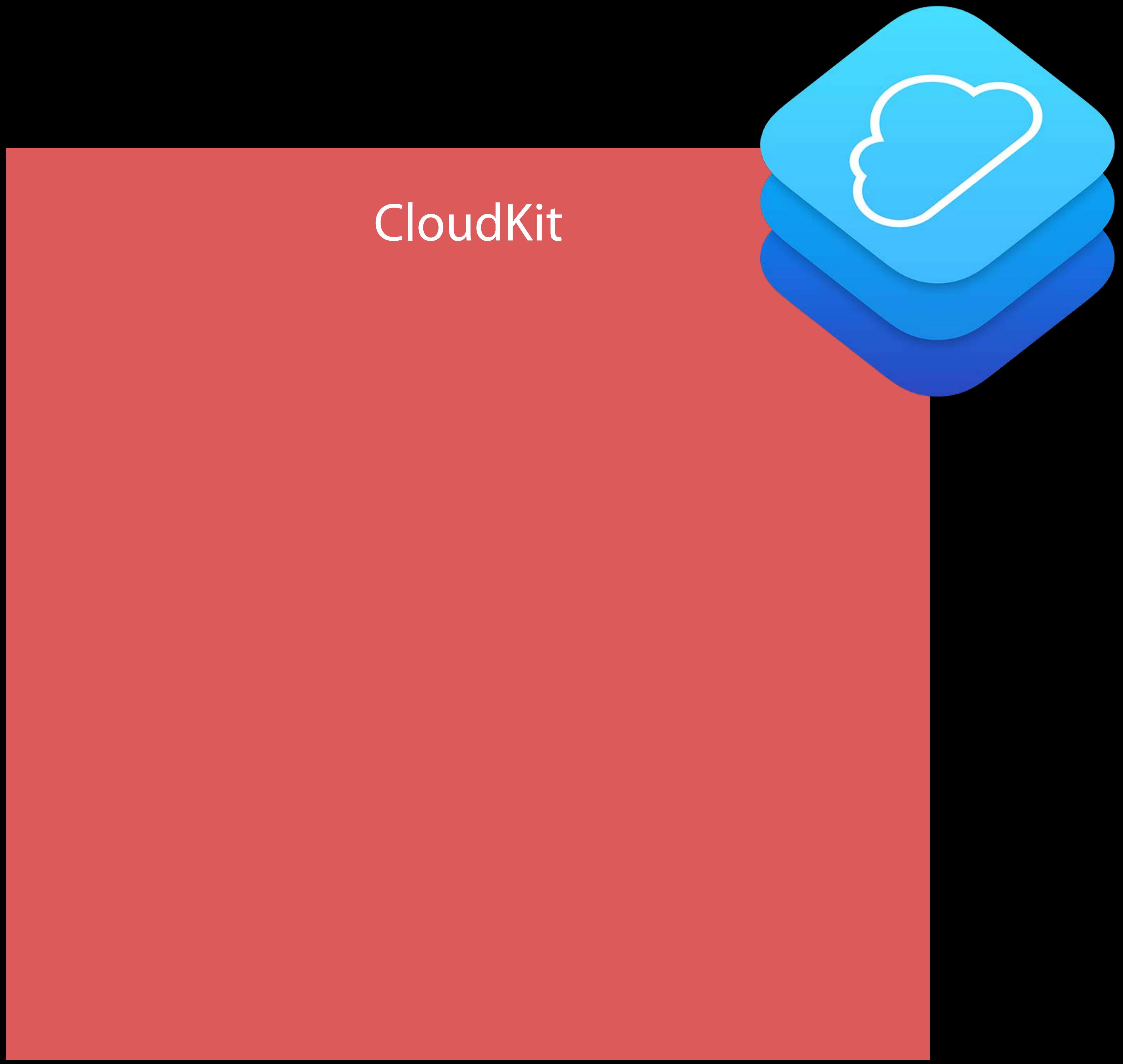
Demo

System Sharing UI

CloudKit Sharing UI

Where does it live?

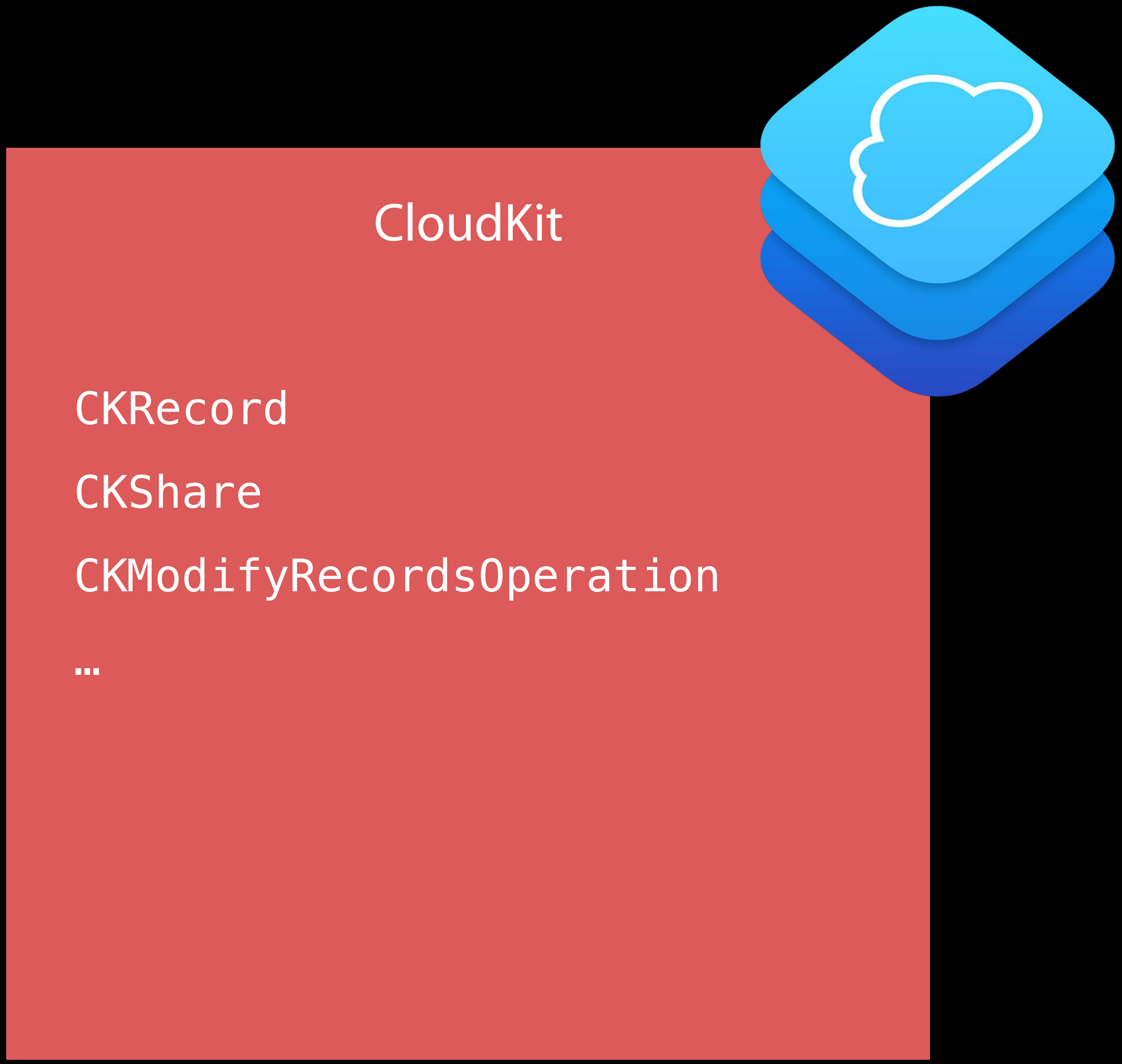
NEW



NEW

CloudKit Sharing UI

Where does it live?



```
CKRecord
CKShare
CKModifyRecordsOperation
…
```

CloudKit Sharing UI

Where does it live?

NEW

CloudKit

```
CKRecord
CKShare
CKModifyRecordsOperation
…
```

macOS

AppKit

```
NSSharingService
NSItemProvider
```

iOS

UIKit

```
UICloudSharingController
```

iOS Sharing API

UICloudSharingController

```swift
// Create a CloudKit share record
let share = CKShare(rootRecord: rootRecord)
share[CKShareTitleKey] = "Shopping List”
share[CKShareThumbnailImageDataKey] = shoppingListThumbnail
```

```swift
// Create a CloudKit share record
let share = CKShare(rootRecord: rootRecord)
share[CKShareTitleKey] = "Shopping List”
share[CKShareThumbnailImageDataKey] = shoppingListThumbnail
```

```swift
// Create a CloudKit share record
let share = CKShare(rootRecord: rootRecord)
share[CKShareTitleKey] = "Shopping List”
share[CKShareThumbnailImageDataKey] = shoppingListThumbnail
```

```swift
// Create a CloudKit share record
let share = CKShare(rootRecord: rootRecord)
share[CKShareTitleKey] = "Shopping List”
share[CKShareThumbnailImageDataKey] = shoppingListThumbnail
```

```
// Create a cloud sharing controller
let sharingController = UICloudSharingController(share: share) {
      (controller: UICloudSharingController,
       prepareCompletionHandler : (CKShare?, CKContainer?, NSError?) -> Void) in
```

```
// Create a cloud sharing controller
let sharingController = UICloudSharingController(share: share) {
      (controller: UICloudSharingController,
       prepareCompletionHandler : (CKShare?, CKContainer?, NSError?) -> Void) in
```

```swift
// Save the share
let sharingController = UICloudSharingController(share: share) {
      (controller: UICloudSharingController,
       prepareCompletionHandler : (CKShare?, CKContainer?, NSError?) -> Void) in
   let modifyOp = CKModifyRecordsOperation(recordsToSave: [rootRecord, share],
      recordIDsToDelete: nil)
   modifyOp.modifyRecordsCompletionBlock = { (_, _, error) in
      prepareCompletionHandler(share, ckContainer, error)
   }
   self.container.privateCloudDatabase.add(modifyOp)
}
```

```swift
// Save the share
let sharingController = UICloudSharingController(share: share) {
      (controller: UICloudSharingController,
       prepareCompletionHandler : (CKShare?, CKContainer?, NSError?) -> Void) in
   let modifyOp = CKModifyRecordsOperation(recordsToSave: [rootRecord, share],
      recordIDsToDelete: nil)
   modifyOp.modifyRecordsCompletionBlock = { (_, _, error) in
      prepareCompletionHandler(share, ckContainer, error)
   }
   self.container.privateCloudDatabase.add(modifyOp)
}
```

```
// Save the share
let sharingController = UICloudSharingController(share: share) {
      (controller: UICloudSharingController,
       prepareCompletionHandler : (CKShare?, CKContainer?, NSError?) -> Void) in
   let modifyOp = CKModifyRecordsOperation(recordsToSave: [rootRecord, share],
      recordIDsToDelete: nil)
   modifyOp.modifyRecordsCompletionBlock = { (_, _, error) in
      prepareCompletionHandler(share, ckContainer, error)
   }
   self.container.privateCloudDatabase.add(modifyOp)
}
```

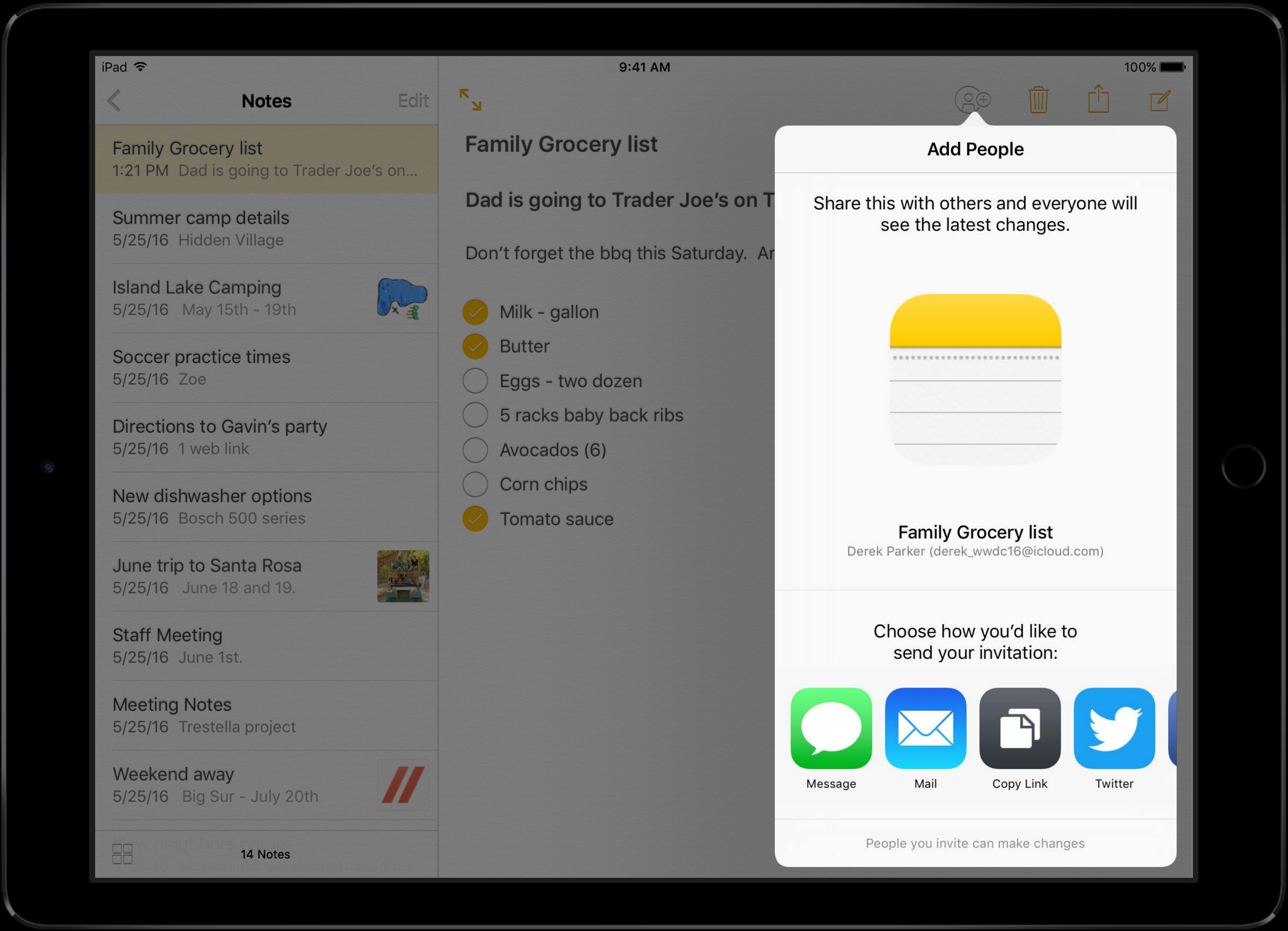
```
// Set sharing options
sharingController.availablePermissions = [.publicOnly, .readWrite]
sharingController.popoverPresentationController?.sourceView = myShareButton
sharingController.delegate = self
self.present(sharingController, animated:true, completion:nil)
```

```
// Set sharing options
sharingController.availablePermissions = [.publicOnly, .readWrite]
sharingController.popoverPresentationController?.sourceView = myShareButton
sharingController.delegate = self
self.present(sharingController, animated:true, completion:nil)
```

```swift
// Set sharing options
sharingController.availablePermissions = [.publicOnly, .readWrite]
sharingController.popoverPresentationController?.sourceView = myShareButton
sharingController.delegate = self
self.present(sharingController, animated:true, completion:nil)
```

```swift
// Set sharing options
sharingController.availablePermissions = [.publicOnly, .readWrite]
sharingController.popoverPresentationController?.sourceView = myShareButton
sharingController.delegate = self
self.present(sharingController, animated:true, completion:nil)
```

```
// Set sharing options
sharingController.availablePermissions = [.publicOnly, .readWrite]
sharingController.popoverPresentationController?.sourceView = myShareButton
sharingController.delegate = self
self.present(sharingController, animated:true, completion:nil)
```



iPad 9:41 AM 100%

Notes Edit

Family Grocery list
1:21 PM Dad is going to Trader Joe's on...

Summer camp details
5/25/16 Hidden Village

Island Lake Camping
5/25/16 May 15th - 19th

Soccer practice times
5/25/16 Zoe

Directions to Gavin's party
5/25/16 1 web link

New dishwasher options
5/25/16 Bosch 500 series

June trip to Santa Rosa
5/25/16 June 18 and 19.

Staff Meeting
5/25/16 June 1st.

Meeting Notes
5/25/16 Trestella project

Weekend away
5/25/16 Big Sur - July 20th

14 Notes

Family Grocery list

Dad is going to Trader Joe's on T

Don't forget the bbq this Saturday. Ar

- ✓ Milk - gallon
- ✓ Butter
- Eggs - two dozen
- 5 racks baby back ribs
- Avocados (6)
- Corn chips
- ✓ Tomato sauce

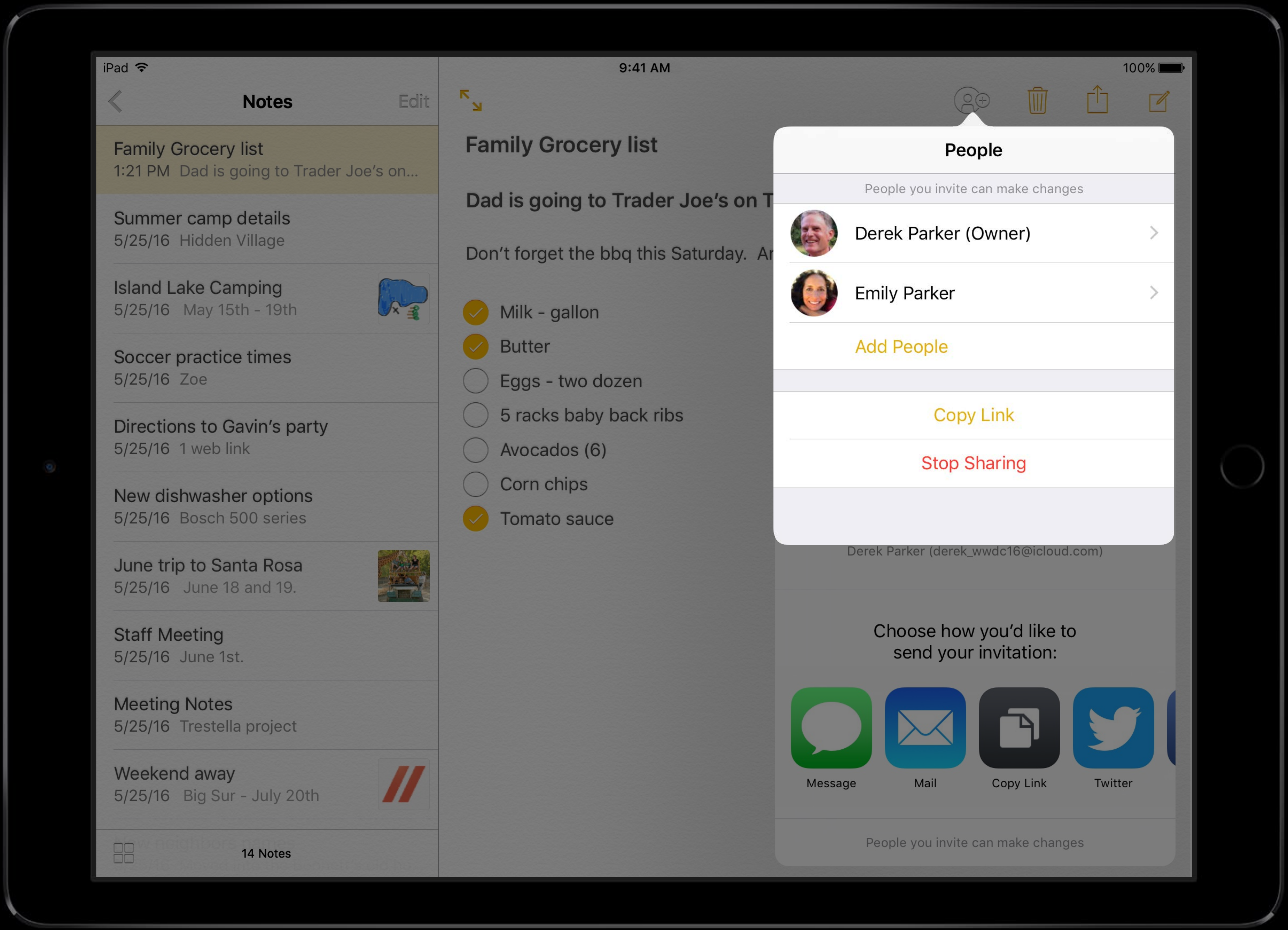
Add People

Share this with others and everyone will see the latest changes.

Family Grocery list
Derek Parker (derek_wwdc16@icloud.com)

Choose how you'd like to send your invitation:

People you invite can make changes



iPad 9:41 AM 100%

Notes Edit

Family Grocery list
1:21 PM Dad is going to Trader Joe's on...

Summer camp details
5/25/16 Hidden Village

Island Lake Camping
5/25/16 May 15th - 19th

Soccer practice times
5/25/16 Zoe

Directions to Gavin's party
5/25/16 1 web link

New dishwasher options
5/25/16 Bosch 500 series

June trip to Santa Rosa
5/25/16 June 18 and 19.

Staff Meeting
5/25/16 June 1st.

Meeting Notes
5/25/16 Trestella project

Weekend away
5/25/16 Big Sur - July 20th

14 Notes

Family Grocery list

Dad is going to Trader Joe's on T

Don't forget the bbq this Saturday. Ar

- ✓ Milk - gallon
- ✓ Butter
- Eggs - two dozen
- 5 racks baby back ribs
- Avocados (6)
- Corn chips
- ✓ Tomato sauce

People

People you invite can make changes

Derek Parker (Owner)

Emily Parker

Add People

Copy Link

Stop Sharing

Derek Parker (derek_wwdc16@icloud.com)

Choose how you'd like to send your invitation:

Message Mail Copy Link Twitter

People you invite can make changes

macOS Sharing API

NSSharingService

```
// Save the share
let itemProvider = NSItemProvider()
itemProvider.registerCloudKitShare { (prepareCompletionHandler :
      (CKShare?, CKContainer?, NSError?) -> Void) in
   // Save the share and root record
}
let sharingService = NSSharingService(named: NSSharingServiceNameCloudSharing)!
sharingService.delegate = self
sharingService.perform(withItems: [itemProvider])
```

```swift
// Save the share
let itemProvider = NSItemProvider()
itemProvider.registerCloudKitShare { (prepareCompletionHandler :
      (CKShare?, CKContainer?, NSError?) -> Void) in
   // Save the share and root record
}
let sharingService = NSSharingService(named: NSSharingServiceNameCloudSharing)!
sharingService.delegate = self
sharingService.perform(withItems: [itemProvider])
```

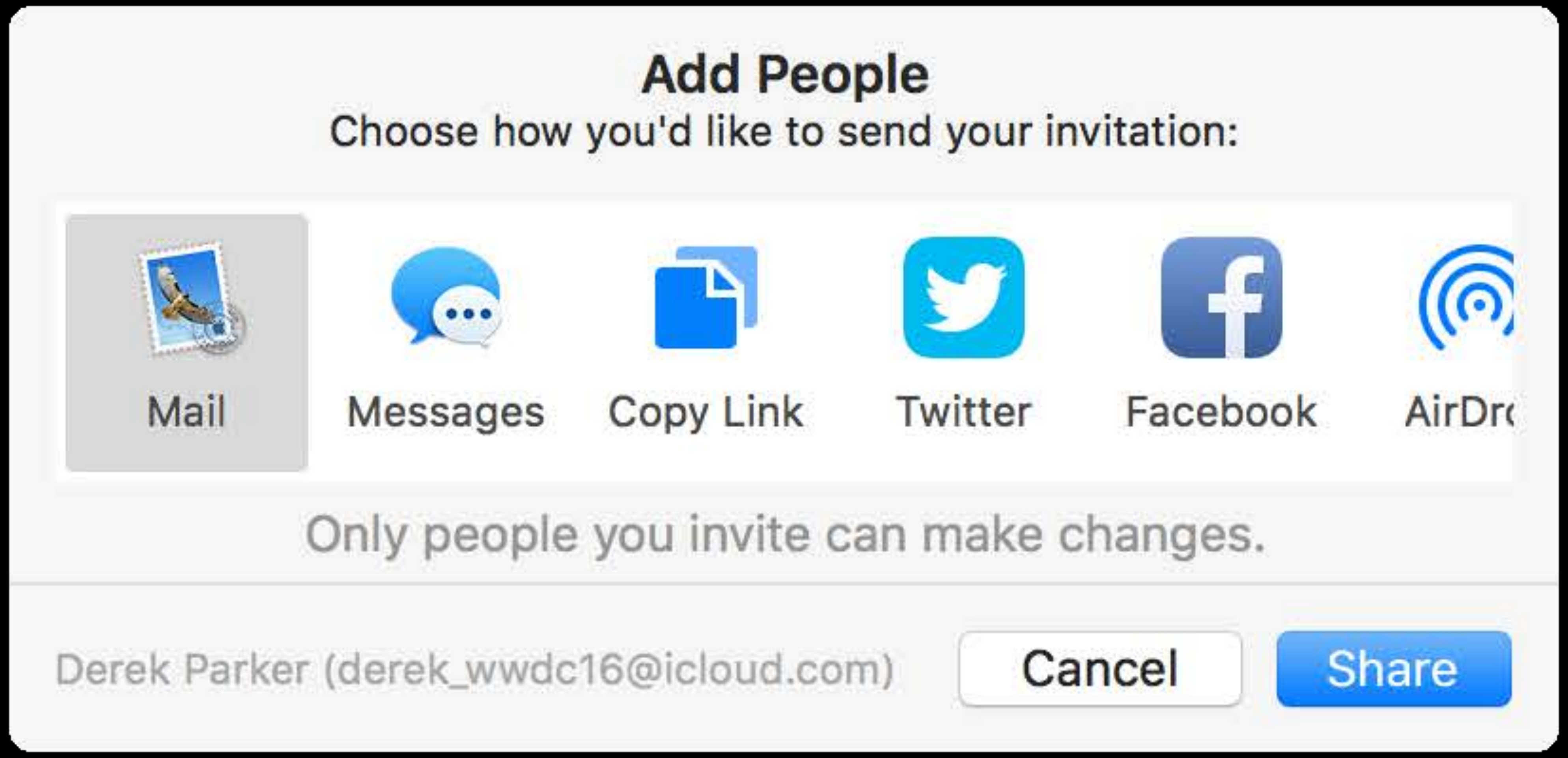
```swift
// Save the share
let itemProvider = NSItemProvider()
itemProvider.registerCloudKitShare { (prepareCompletionHandler :
      (CKShare?, CKContainer?, NSError?) -> Void) in
   // Save the share and root record
}
let sharingService = NSSharingService(named: NSSharingServiceNameCloudSharing)!
sharingService.delegate = self
sharingService.perform(withItems: [itemProvider])
```

```swift
// Save the share
let itemProvider = NSItemProvider()
itemProvider.registerCloudKitShare { (prepareCompletionHandler :
      (CKShare?, CKContainer?, NSError?) -> Void) in
   // Save the share and root record
}
let sharingService = NSSharingService(named: NSSharingServiceNameCloudSharing)!
sharingService.delegate = self
sharingService.perform(withItems: [itemProvider])
```

```
// Save the share
let itemProvider = NSItemProvider()
itemProvider.registerCloudKitShare { (prepareCompletionHandler :
      (CKShare?, CKContainer?, NSError?) -> Void) in
   // Save the share and root record
}
let sharingService = NSSharingService(named: NSSharingServiceNameCloudSharing)!
sharingService.delegate = self
sharingService.perform(withItems: [itemProvider])
```

```swift
// Define sharing options

func options(for: NSSharingService, share: NSItemProvider)-> NSCloudKitSharingServiceOptions
{
   return [.allowPublic, .allowReadWrite]
}
```

```swift
// Define sharing options

func options(for: NSSharingService, share: NSItemProvider)-> NSCloudKitSharingServiceOptions
{
    return [.allowPublic, .allowReadWrite]
}
```

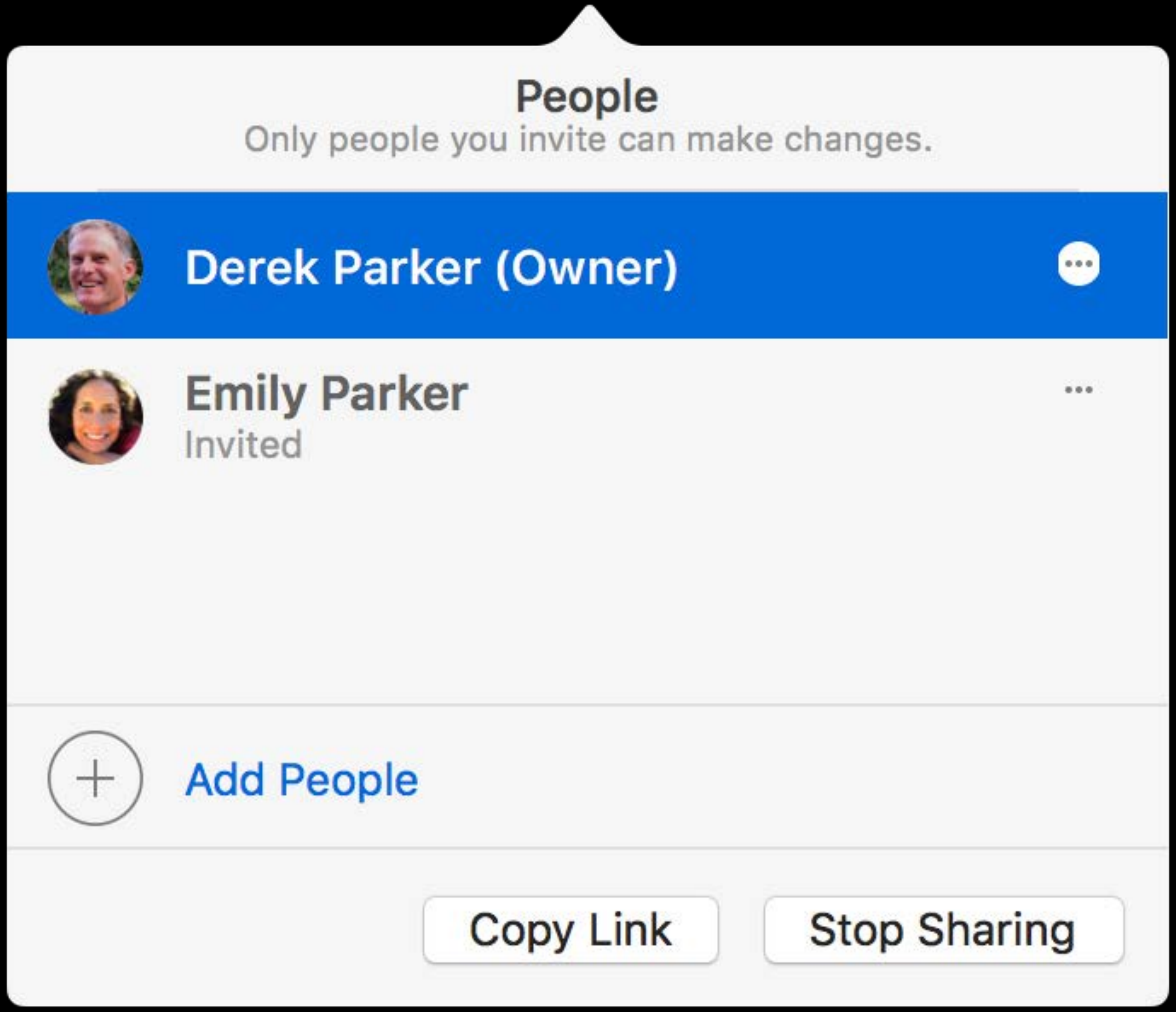


Add People

Choose how you'd like to send your invitation:

Only people you invite can make changes.

Derek Parker (derek_wwdc16@icloud.com)


People
Only people you invite can make changes.
Derek Parker (Owner)
Emily Parker
Invited
Add People
Copy Link
Stop Sharing

```
// User clicked a share
public class NSApplication {
   public func application(application: NSApplication,
         userAcceptedCloudKitShareWith: CKShareMetadata)
}
```

macOS

```
// User clicked a share
public class NSApplication {
   public func application(application: NSApplication,
         userAcceptedCloudKitShareWith: CKShareMetadata)
}

public class UIApplication {
   public func application(application: UIApplication,
         userAcceptedCloudKitShareWith: CKShareMetadata)
}
```

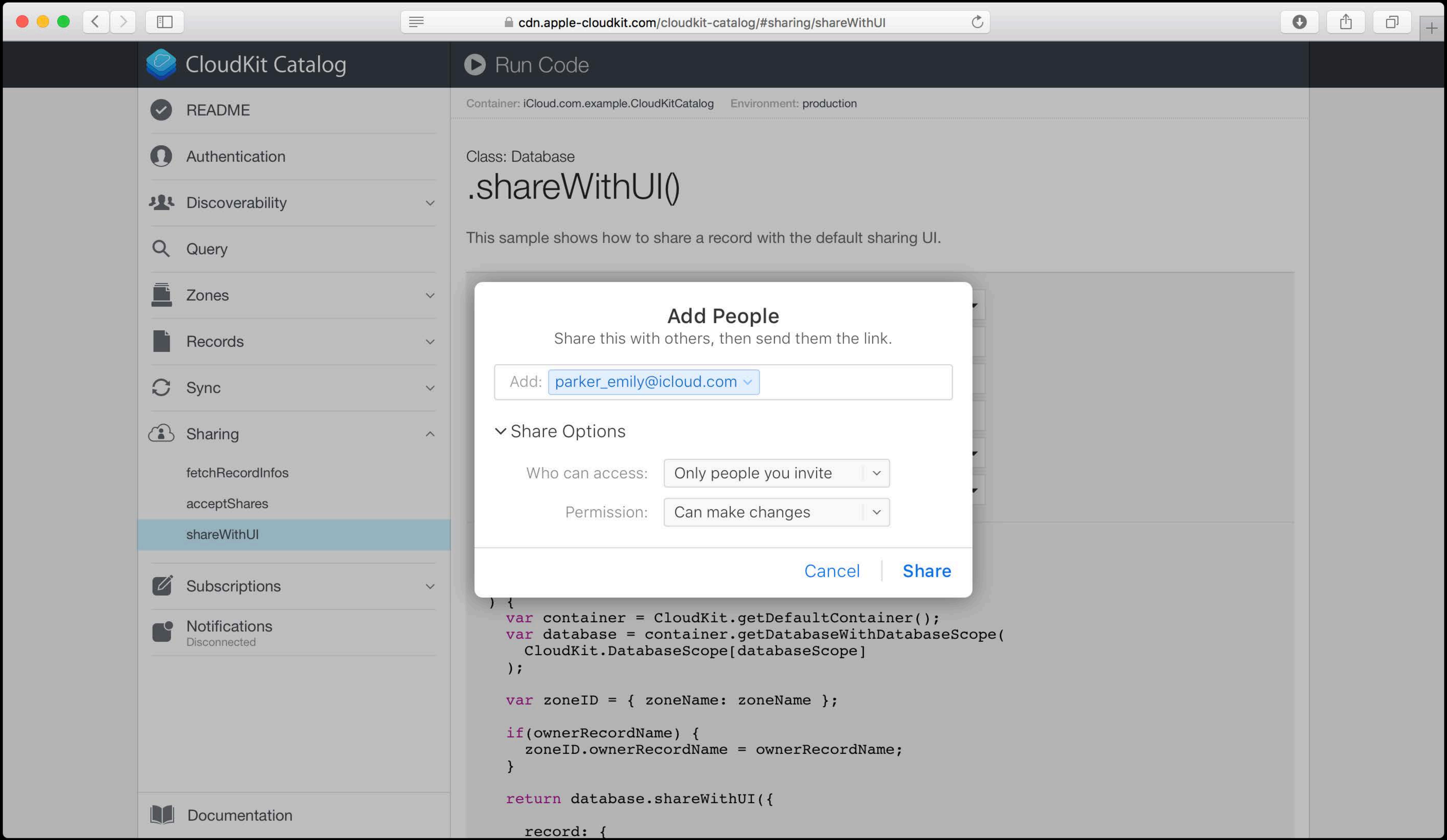
macOS

iOS

```
// Add an Info.plist key for CloudKit Sharing
<key>CKSharingSupported</key>
<true/>
```

CloudKit JS

Web Sharing UI



Container: iCloud.com.example.CloudKitCatalog Environment: production

Class: Database

.shareWithUI()

This sample shows how to share a record with the default sharing UI.

Add People

Share this with others, then send them the link.

Add: parker_emily@icloud.com

Share Options

Who can access: Only people you invite

Permission: Can make changes

```
) {
  var container = CloudKit.getDefaultContainer();
  var database = container.getDatabaseWithDatabaseScope(
    CloudKit.DatabaseScope[databaseScope]
  );

  var zoneID = { zoneName: zoneName };

  if(ownerRecordName) {
    zoneID.ownerRecordName = ownerRecordName;
  }

  return database.shareWithUI({

    record: {
```

Sharing In Depth

Vanessa Hong

Common Use Cases

Deep dive

Sharing multiple records

Zones in shared database

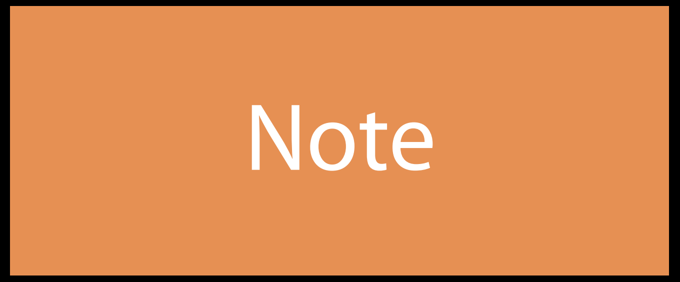
CKShare internals

Sharing APIs

Special notes

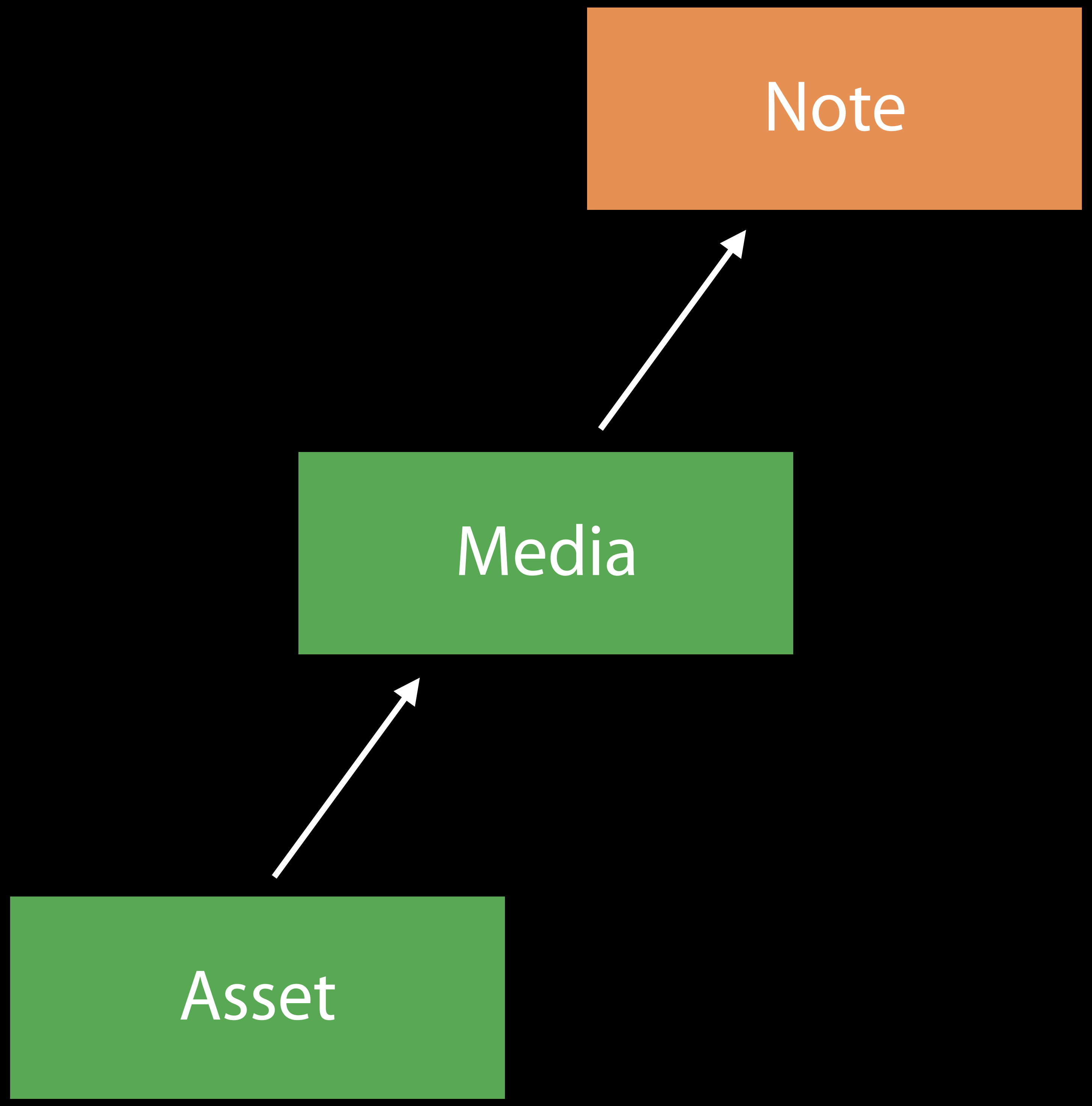
Sharing Multiple Records

A Note is not a single record



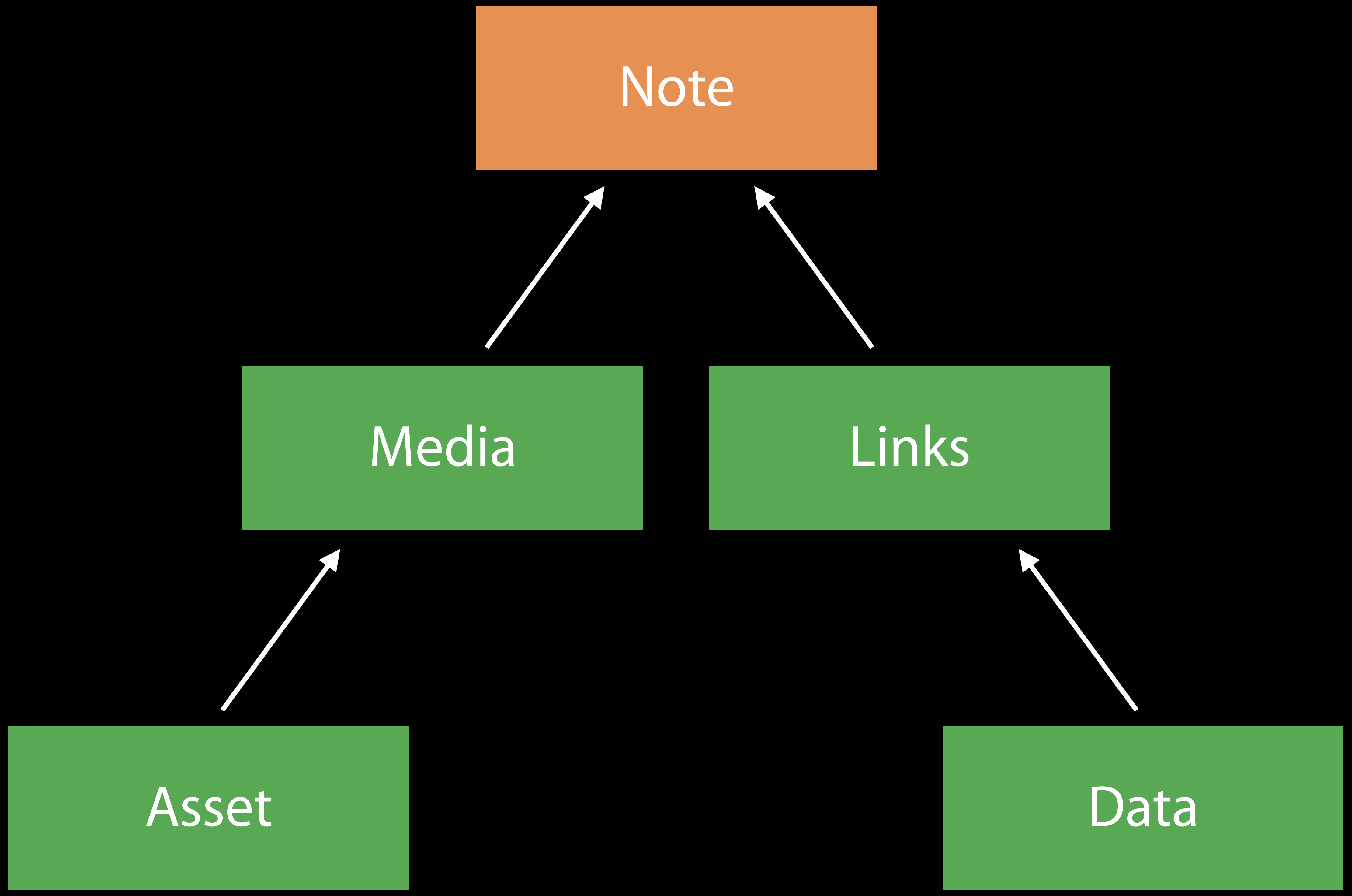
Sharing Multiple Records

A Note consists of many records



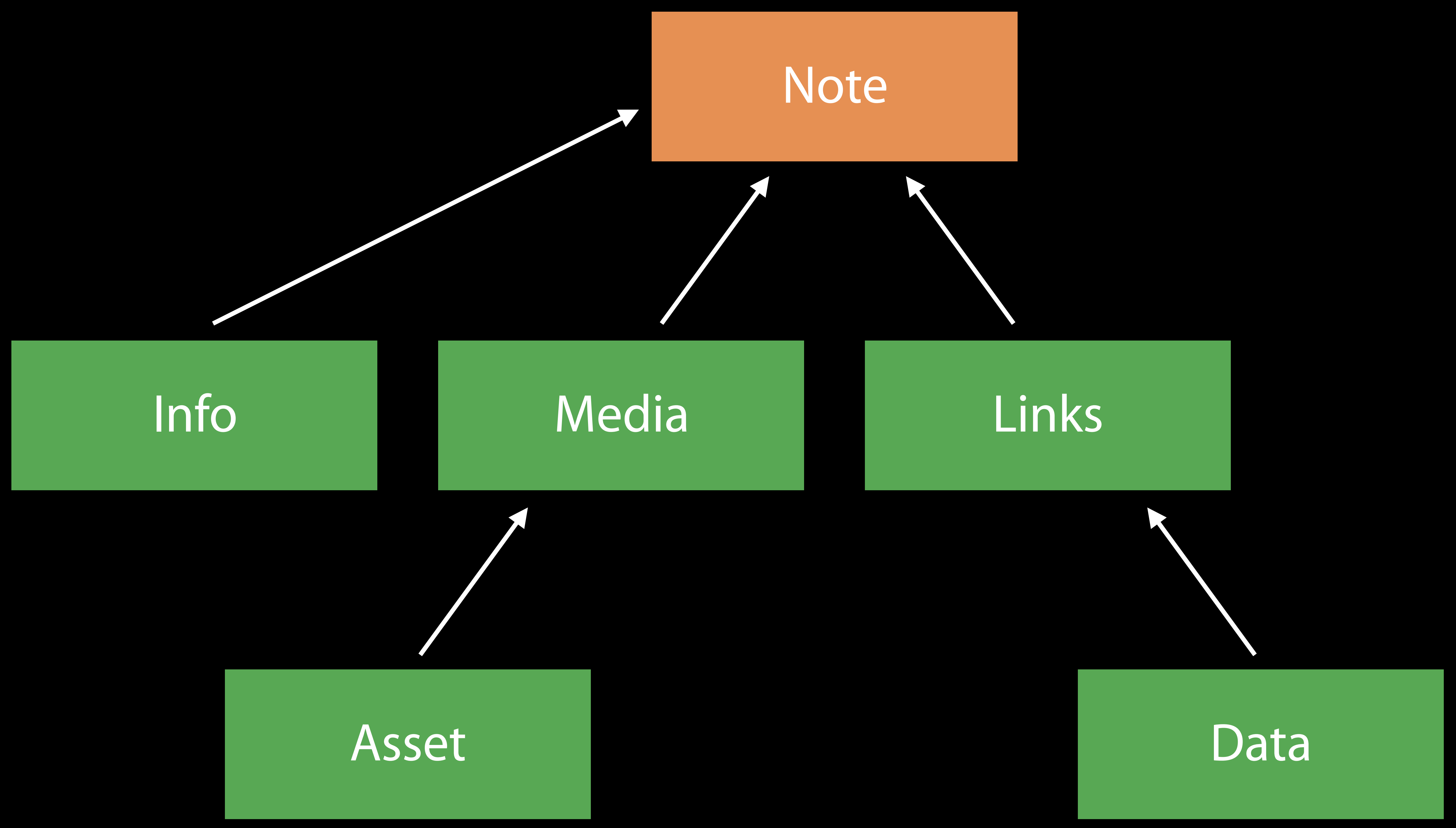
Sharing Multiple Records

A Note consists of many records



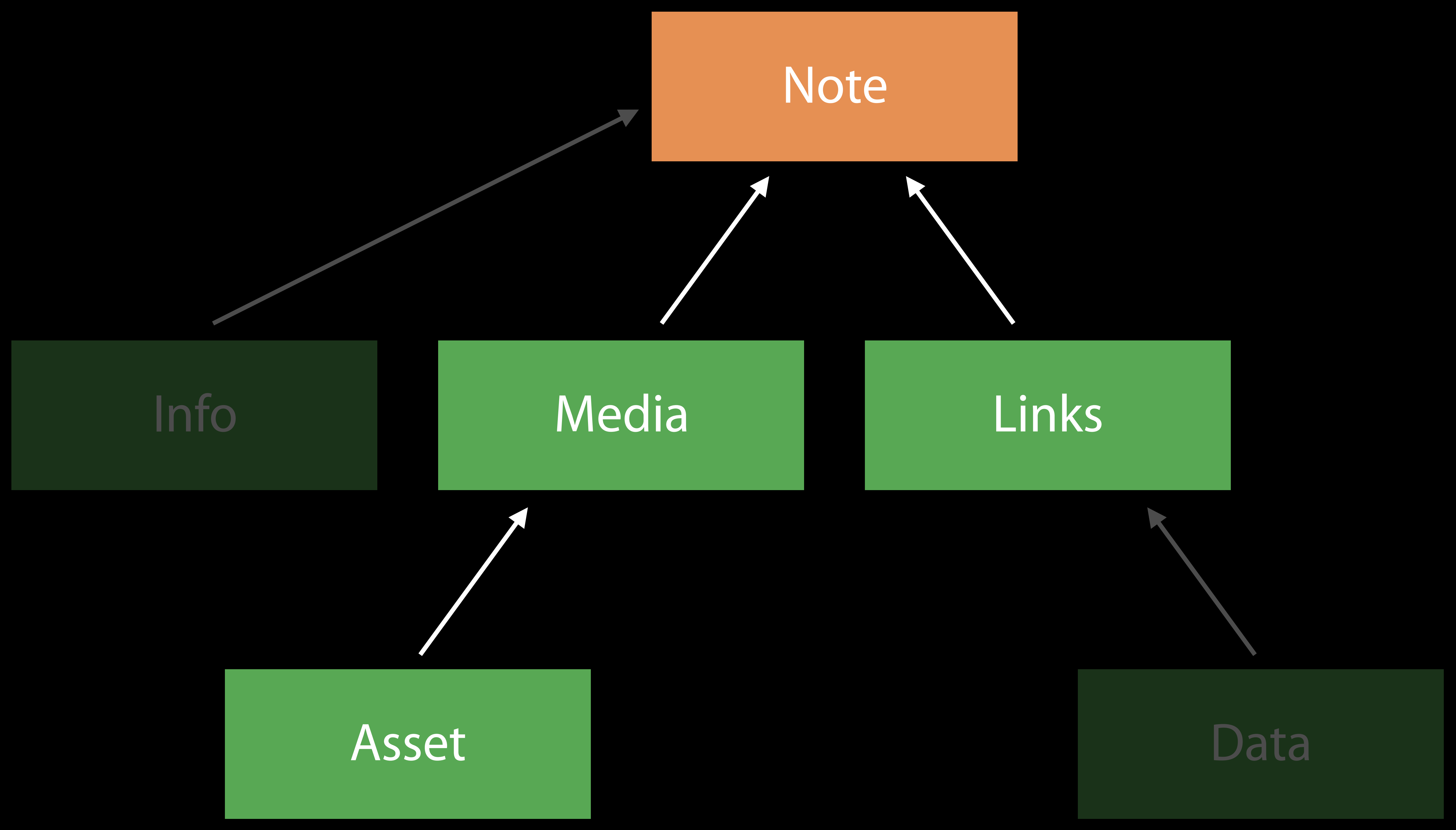
Sharing Multiple Records

A Note consists of many records



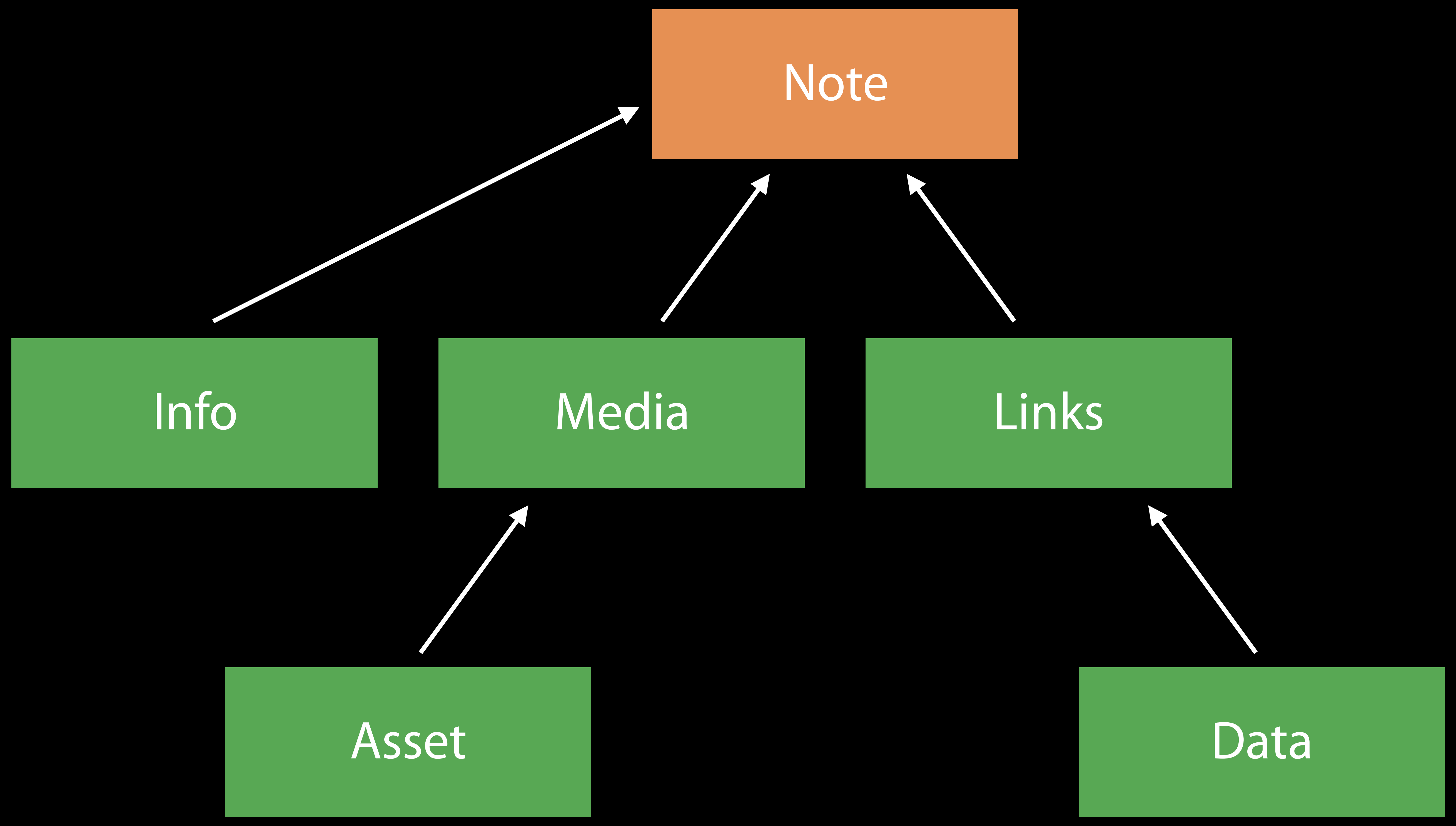
Sharing Multiple Records

Participant should only see a subset



Sharing Multiple Records

Tell us what should be shared



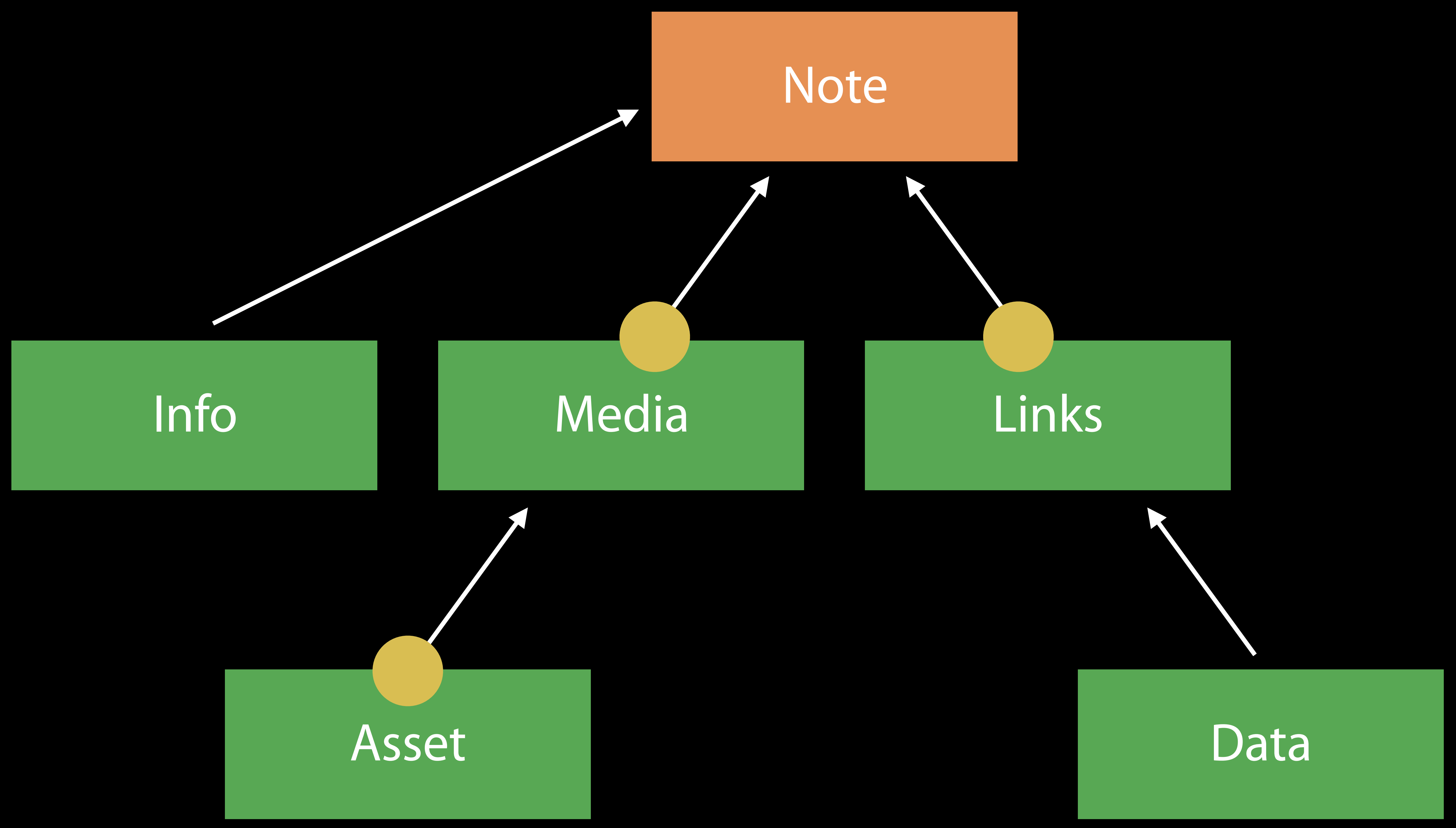
Records have a new property

```
public var parent: CKReference
```

Sharing Multiple Records

Parent references define the hierarchy for sharing

● Parent Field



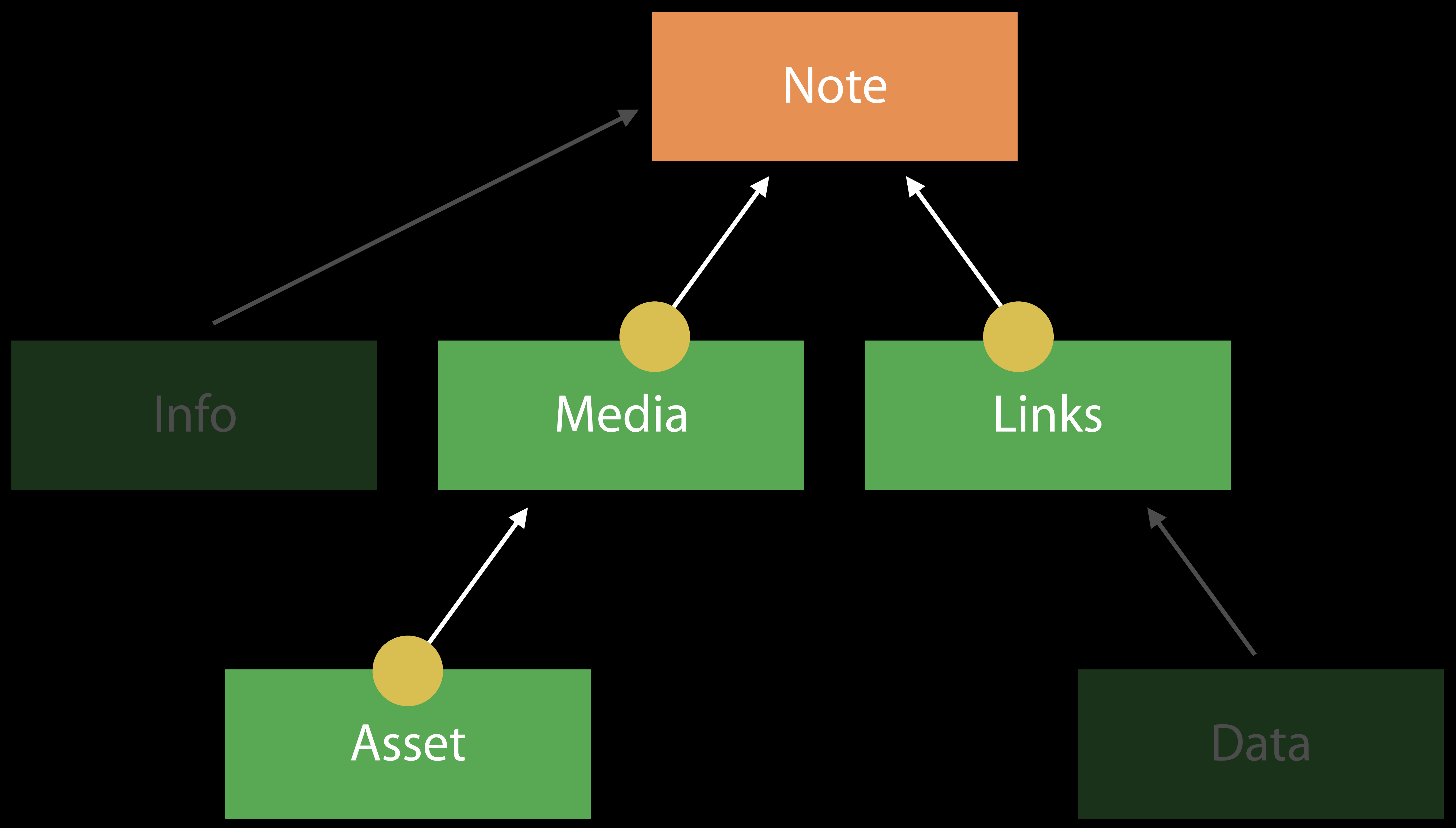
Records have a new property

```
public var parent: CKReference
```

Sharing Multiple Records

● Parent Field

Descendants linked to root record via the parent field



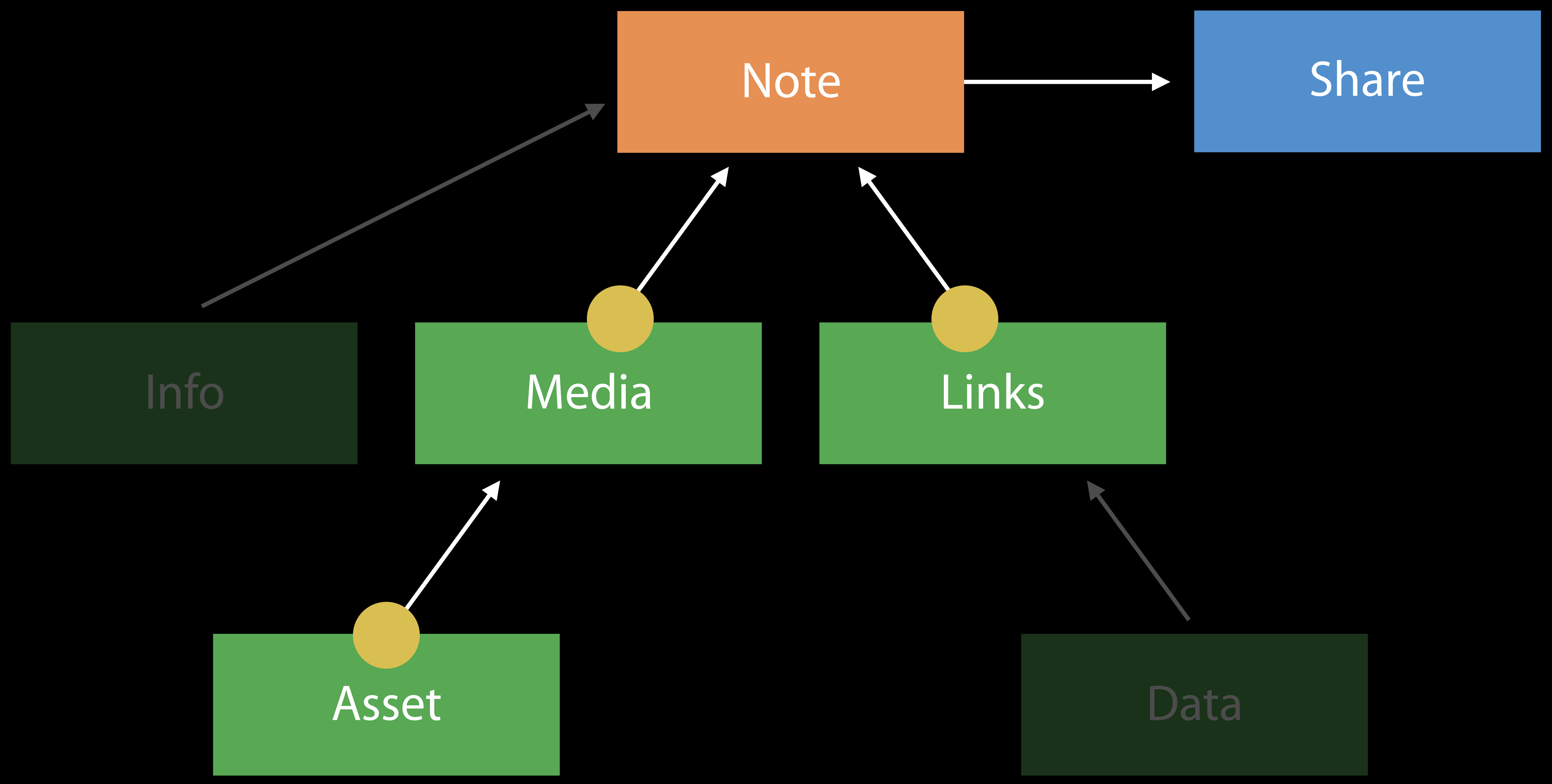
Records have a new property

```
public var parent: CKReference
```

Sharing Multiple Records

Create Share using CKShare(rootRecord:)

● Parent Field

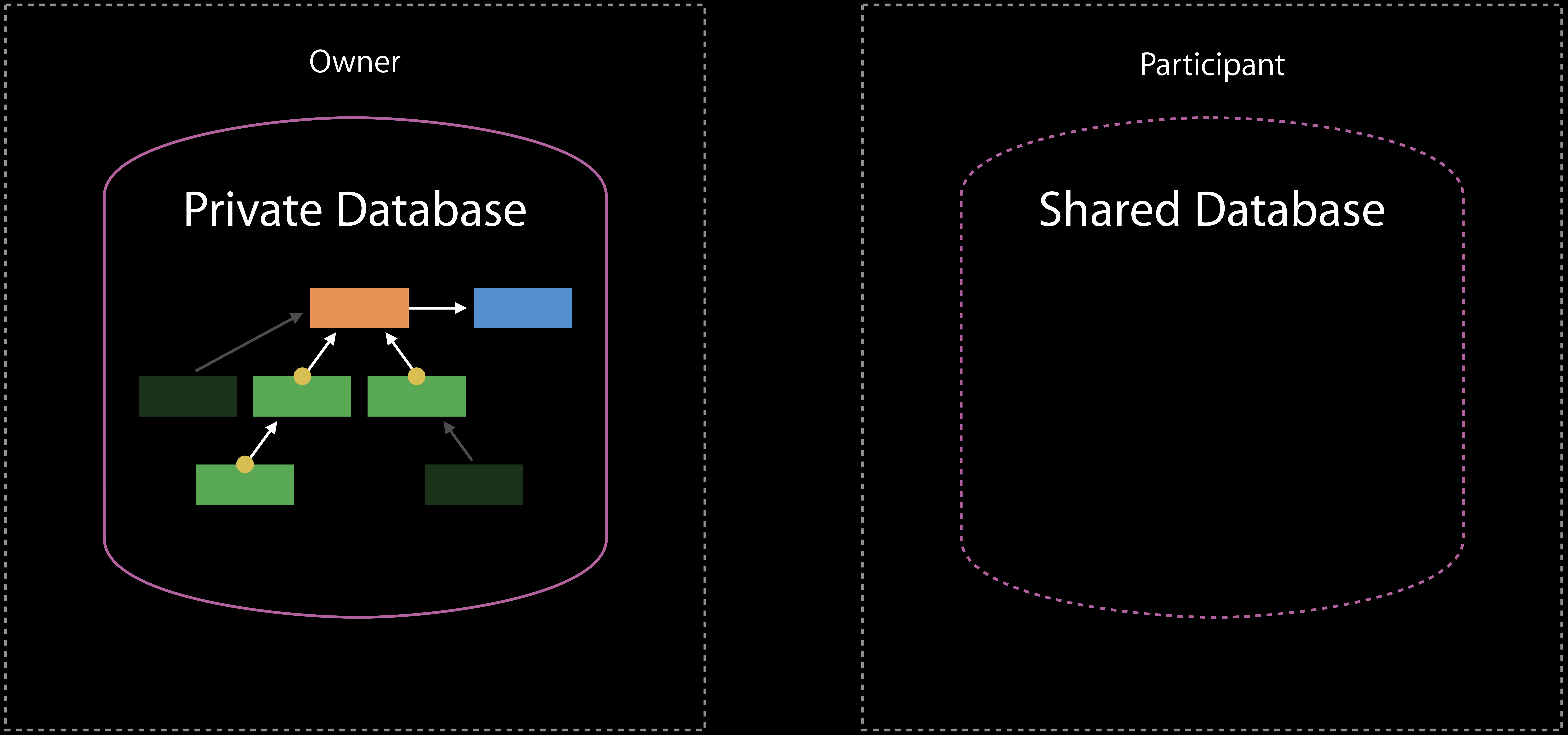


Records have a new property

```
public var parent: CKReference
```

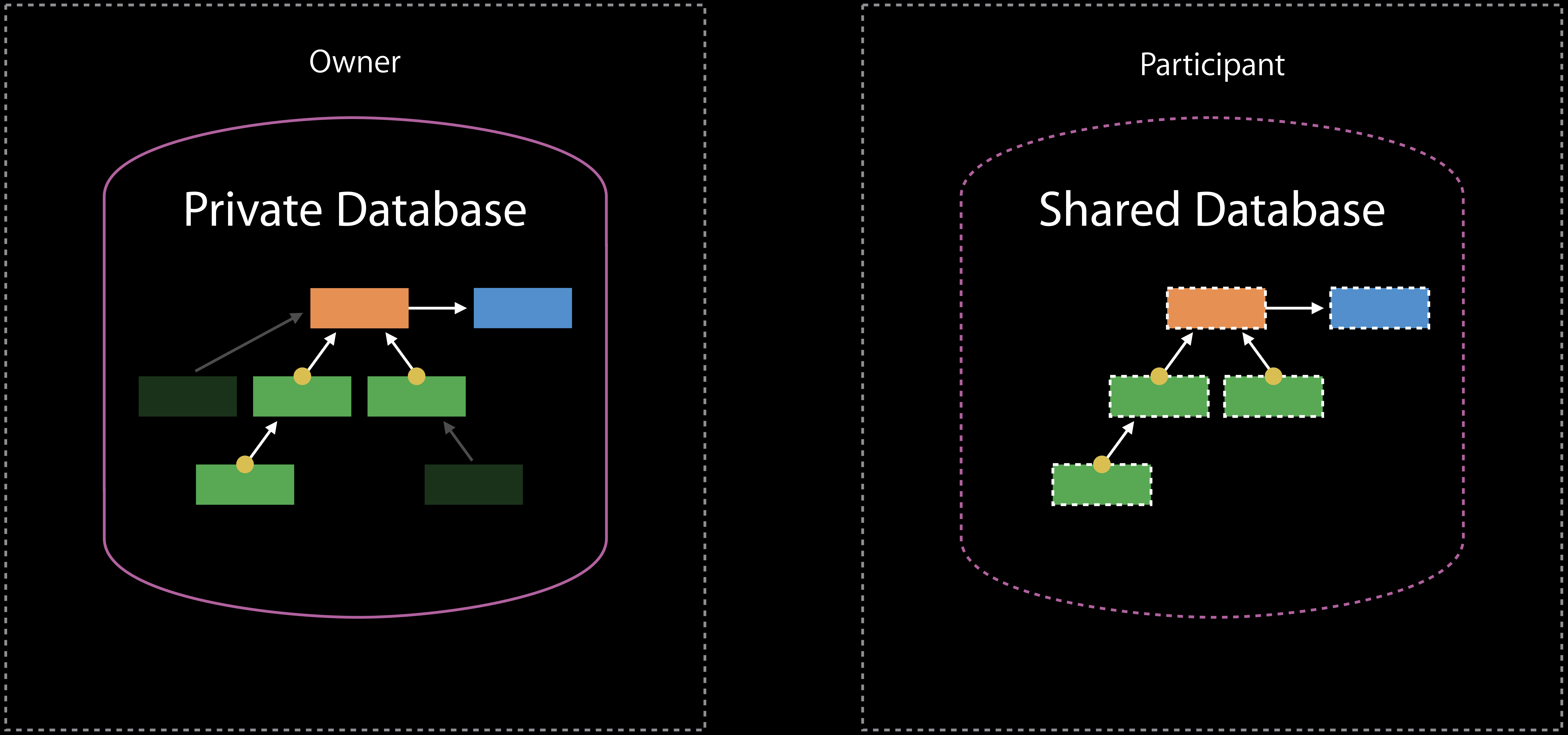
Sharing Multiple Records

Shared DB is only a View into the owner's private DB



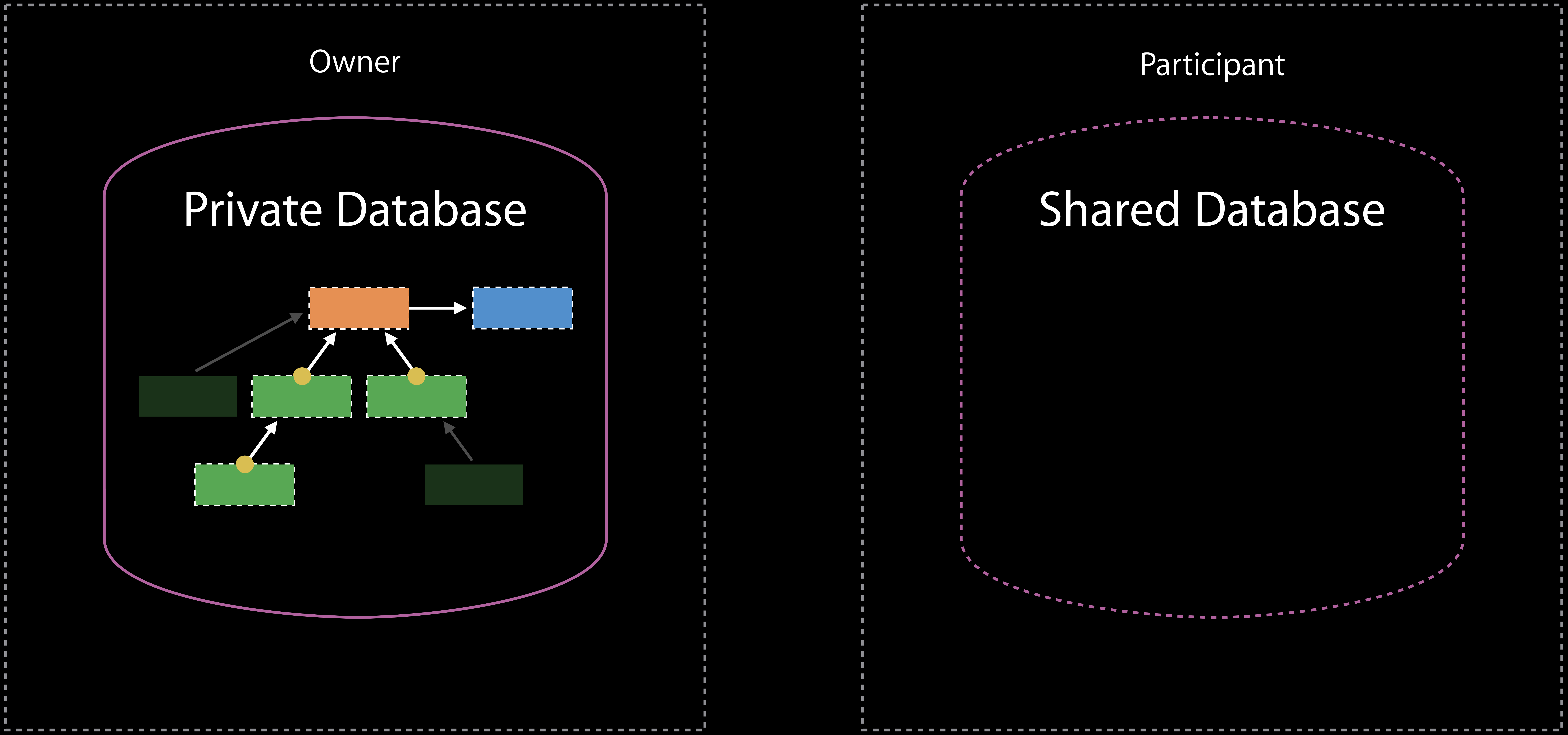
Sharing Multiple Records

After accepting, Participant sees only records with parent



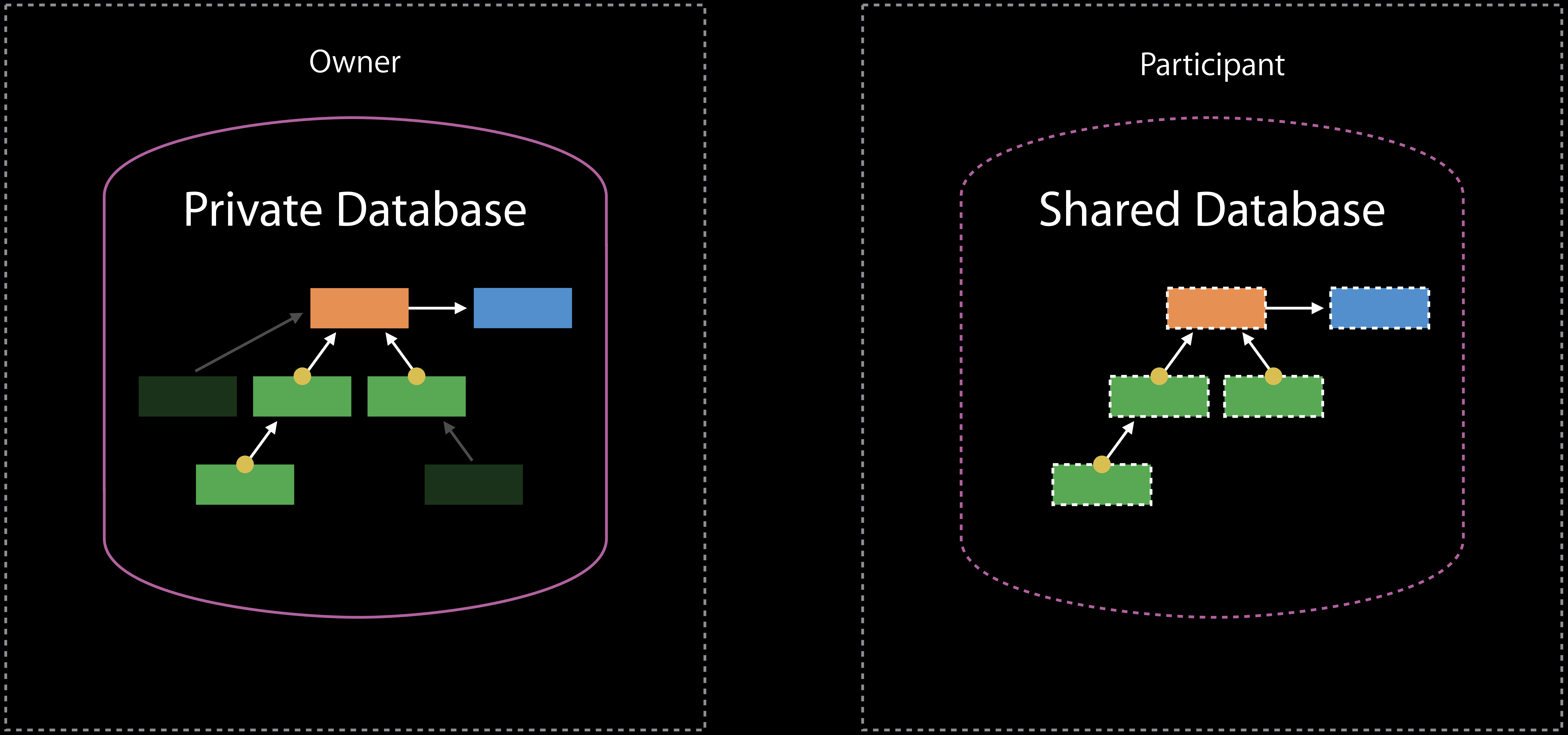
Sharing Multiple Records

After accepting, Participant sees only records with parent



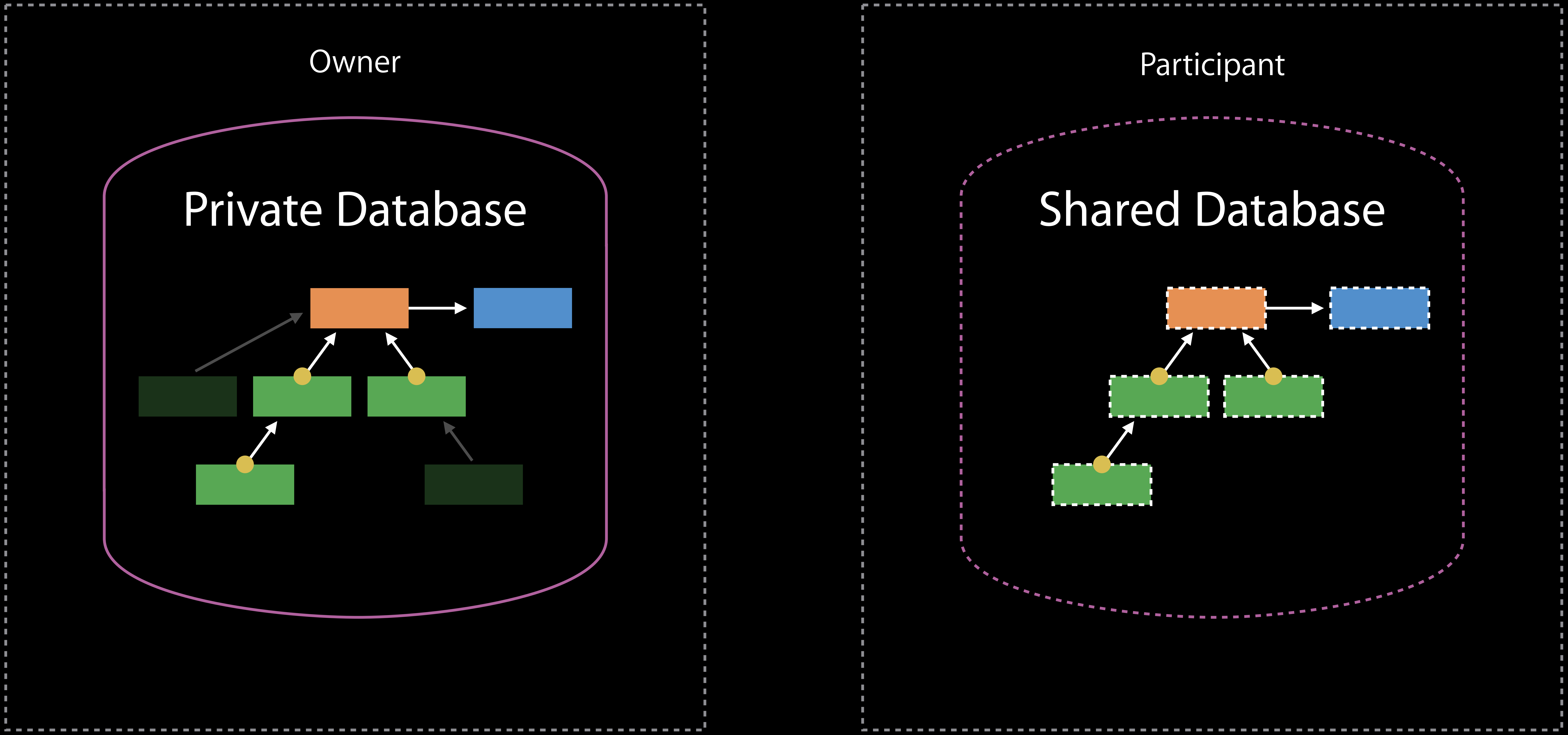
Sharing Multiple Records

After accepting, Participant sees only records with parent



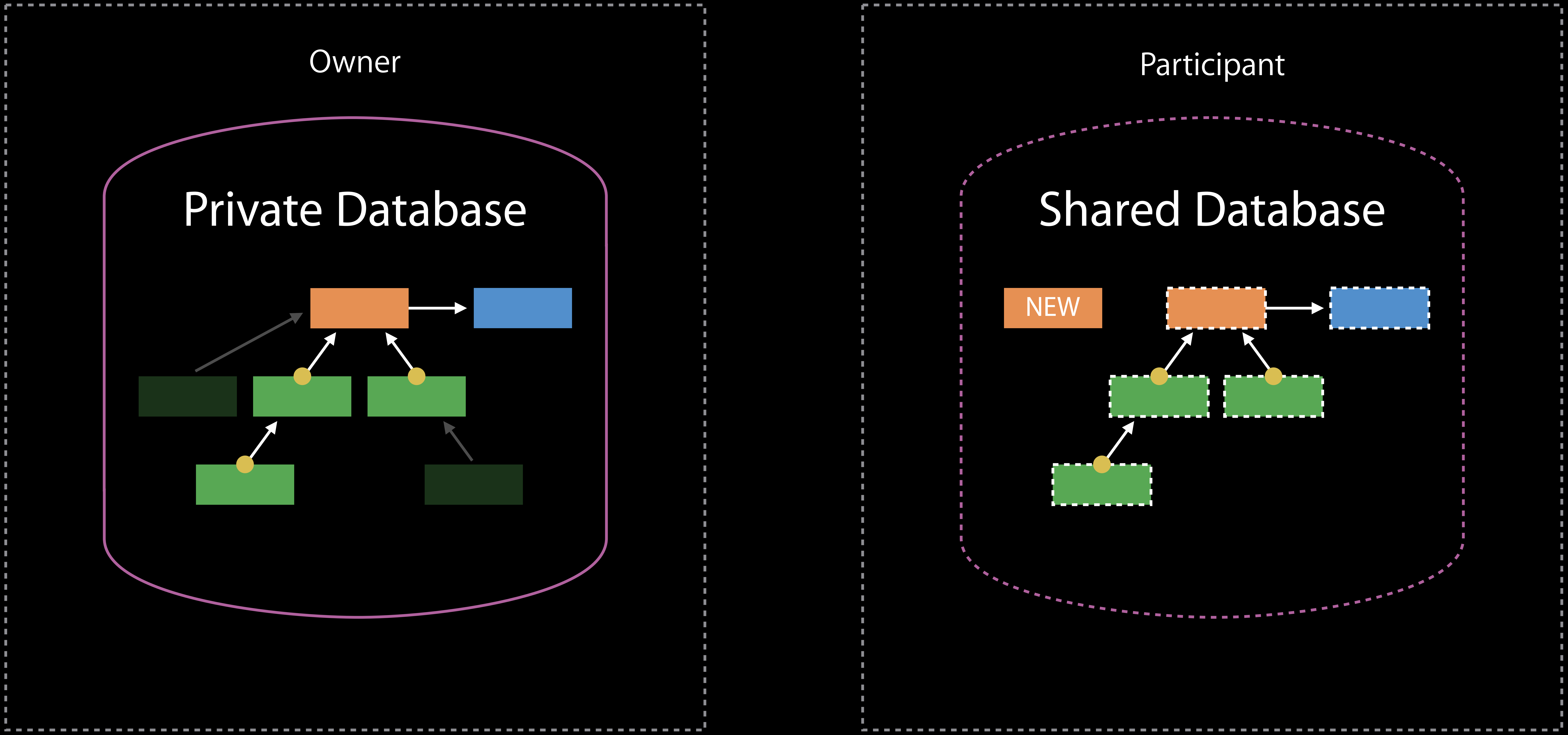
Sharing Multiple Records

readWrite Participant cannot add a dangling CKRecord



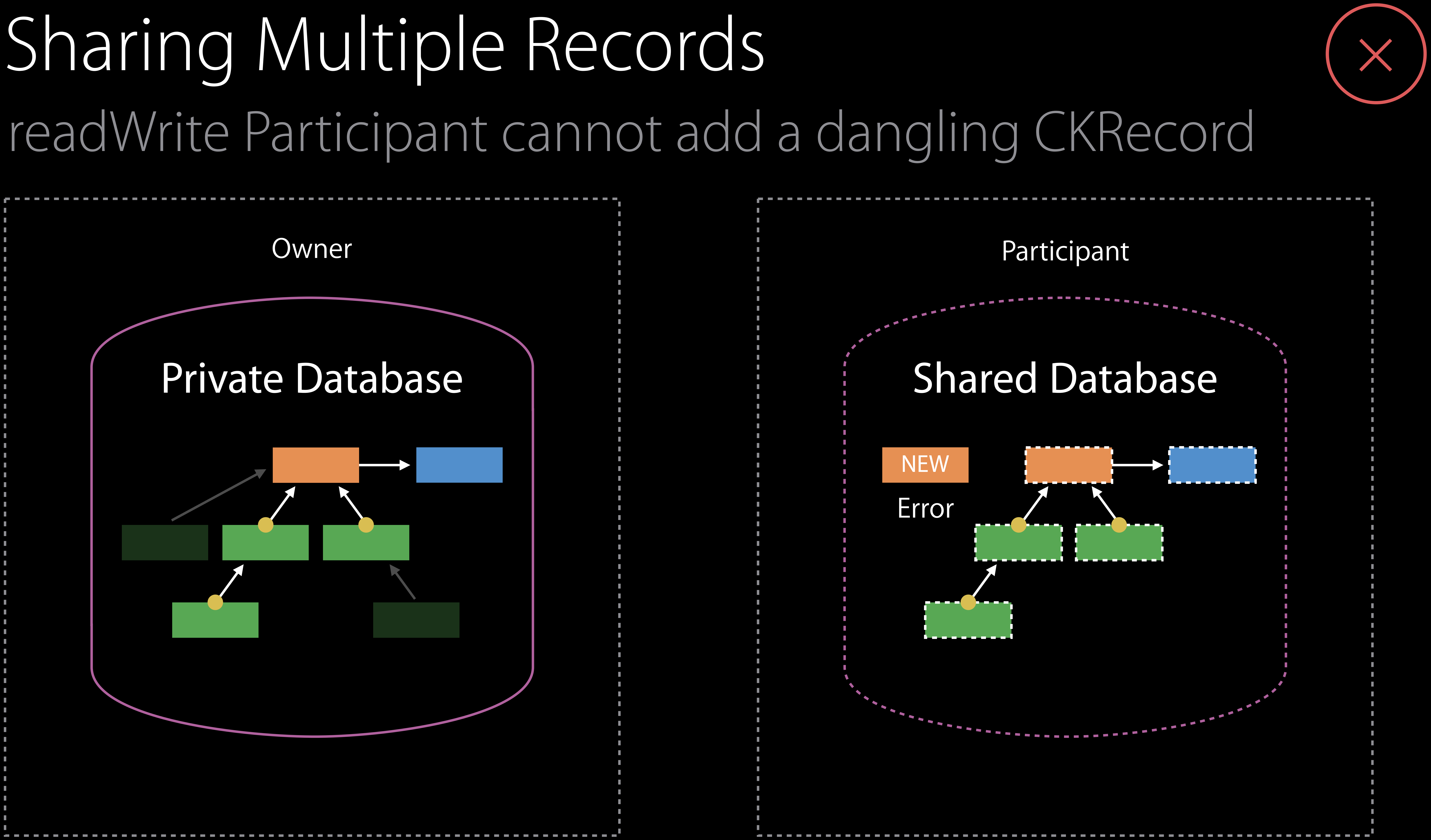
Sharing Multiple Records

readWrite Participant cannot add a dangling CKRecord



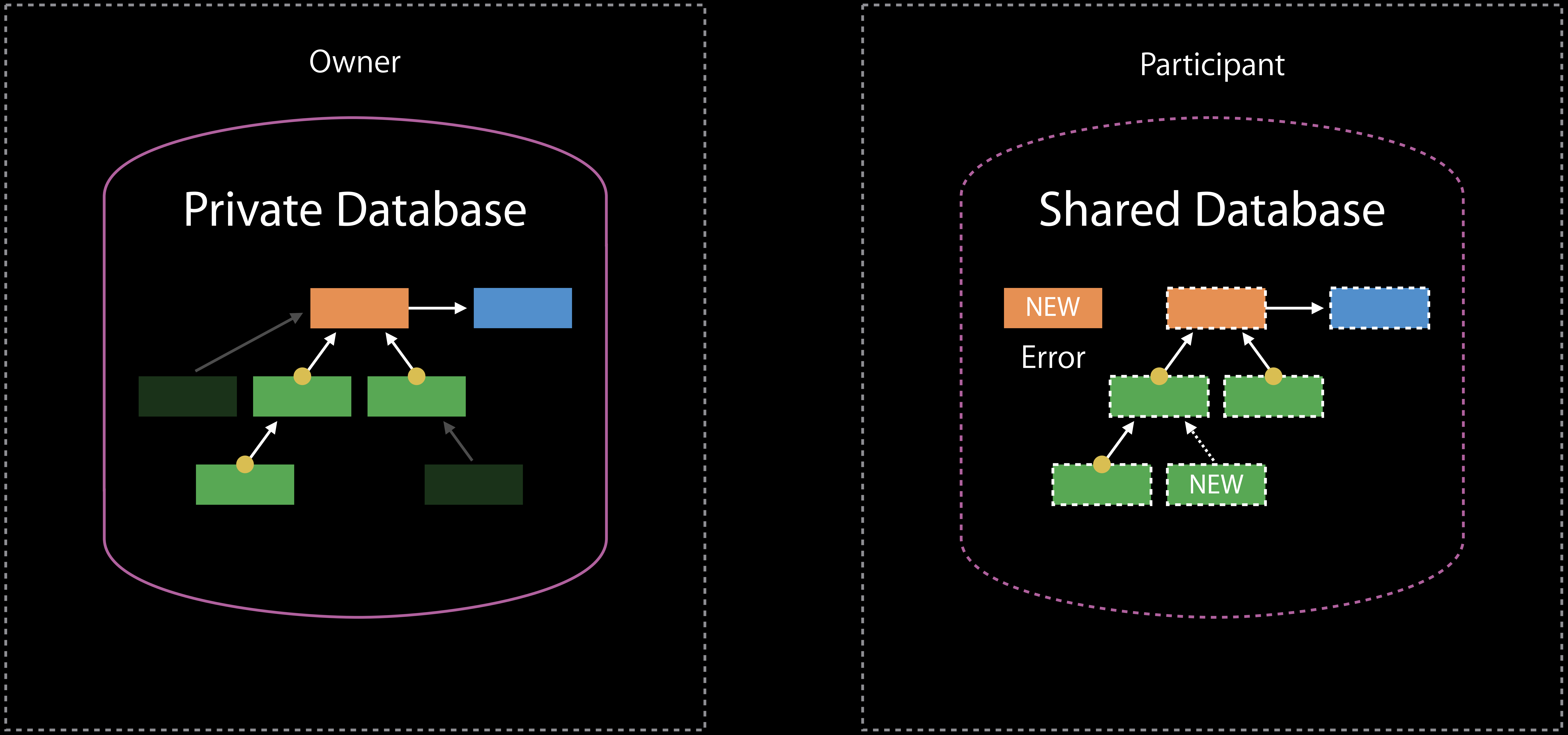
Sharing Multiple Records

readWrite Participant cannot add a dangling CKRecord



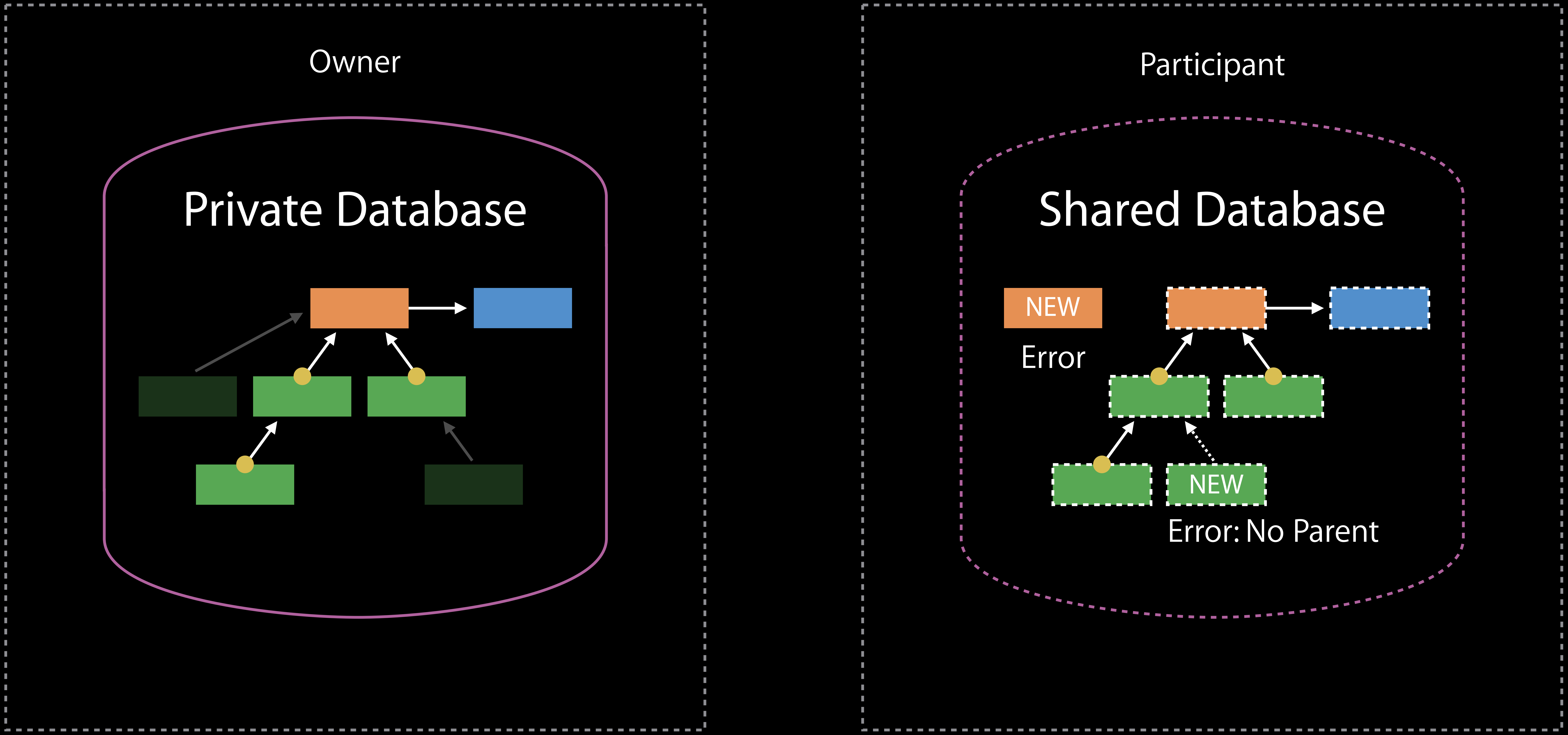
Sharing Multiple Records

readWrite Participant cannot add a dangling CKRecord



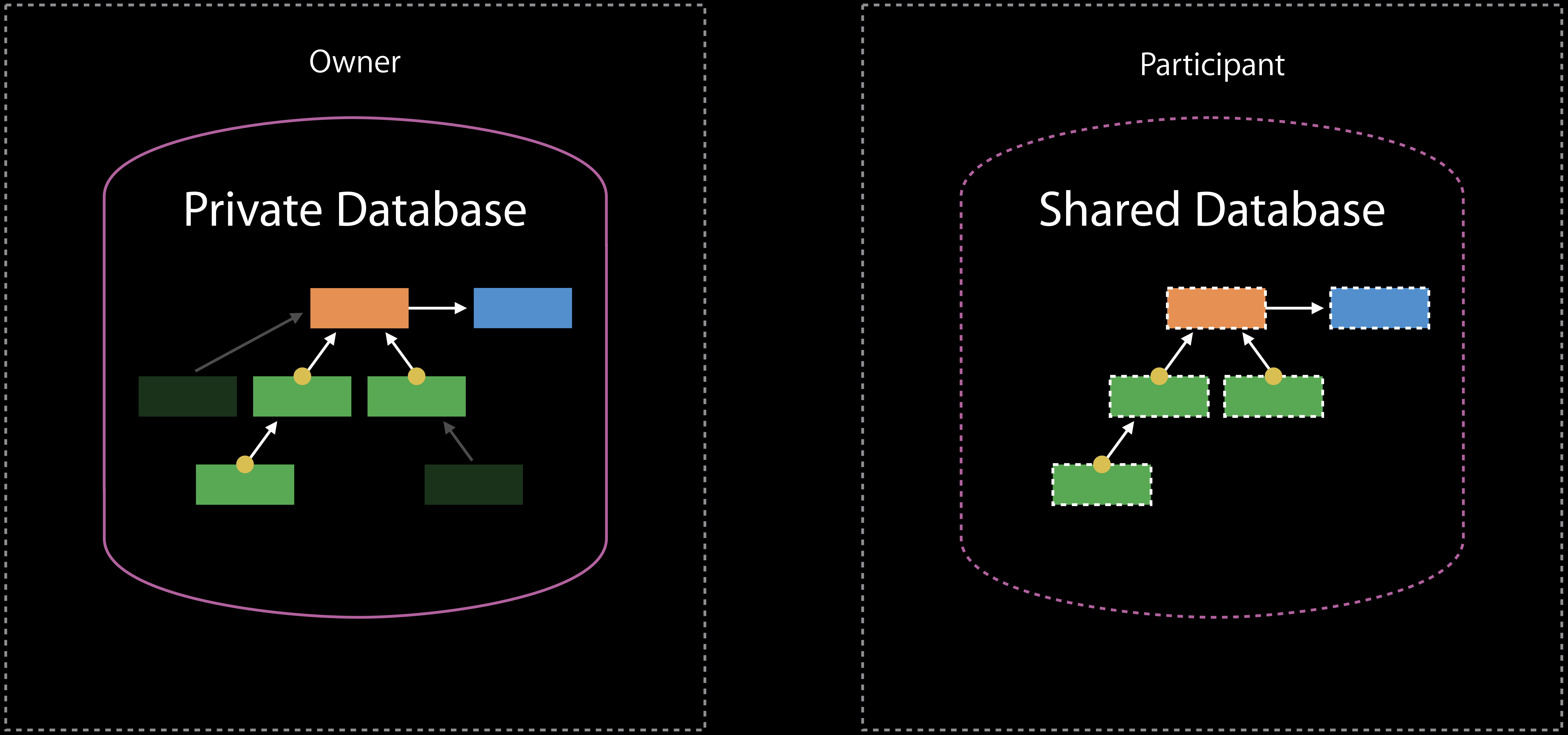
Sharing Multiple Records

readWrite Participant cannot add a dangling CKRecord



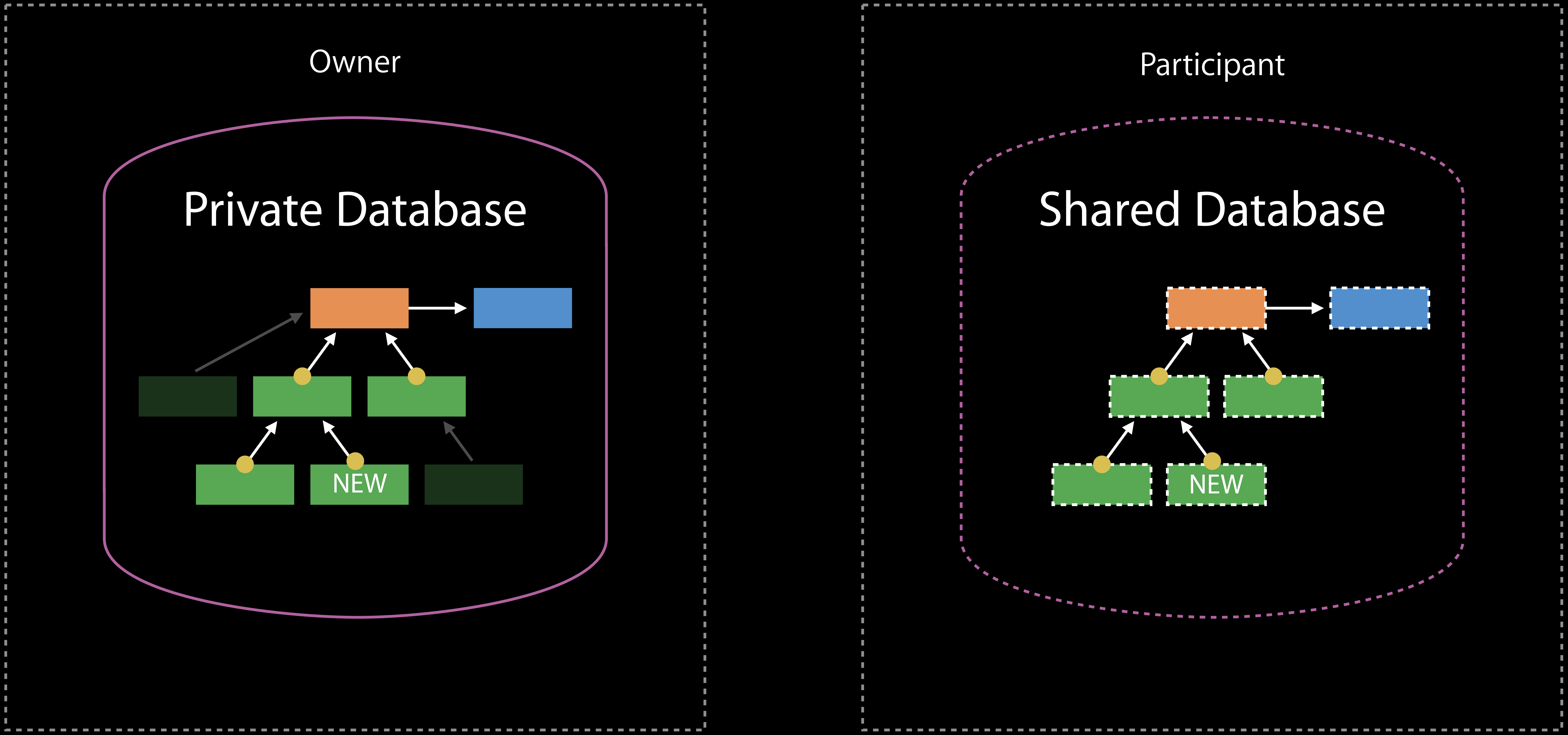
Sharing Multiple Records

readWrite Participant can add a new parented CKRecord



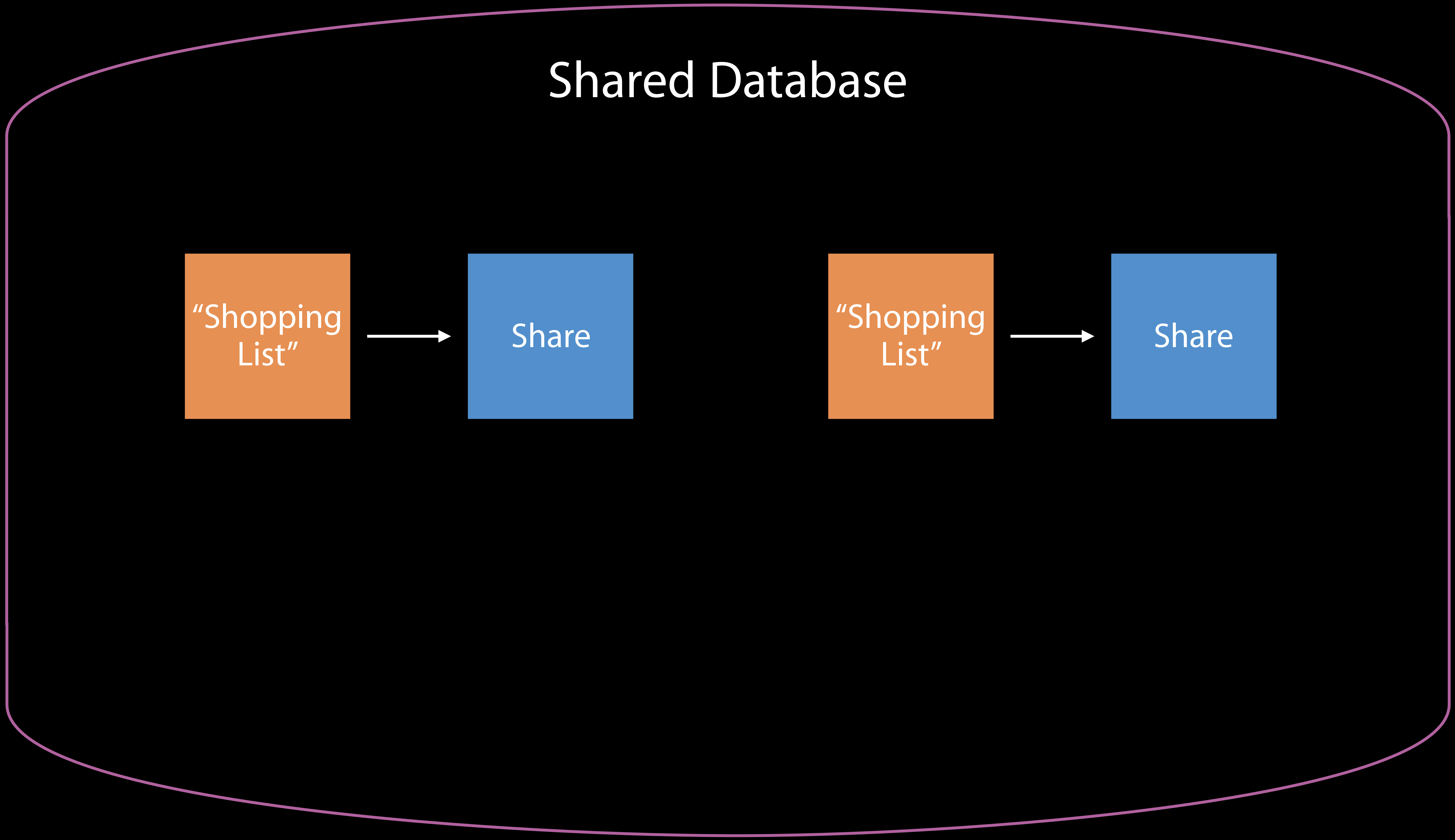
Sharing Multiple Records

readWrite Participant can add a new parented CKRecord



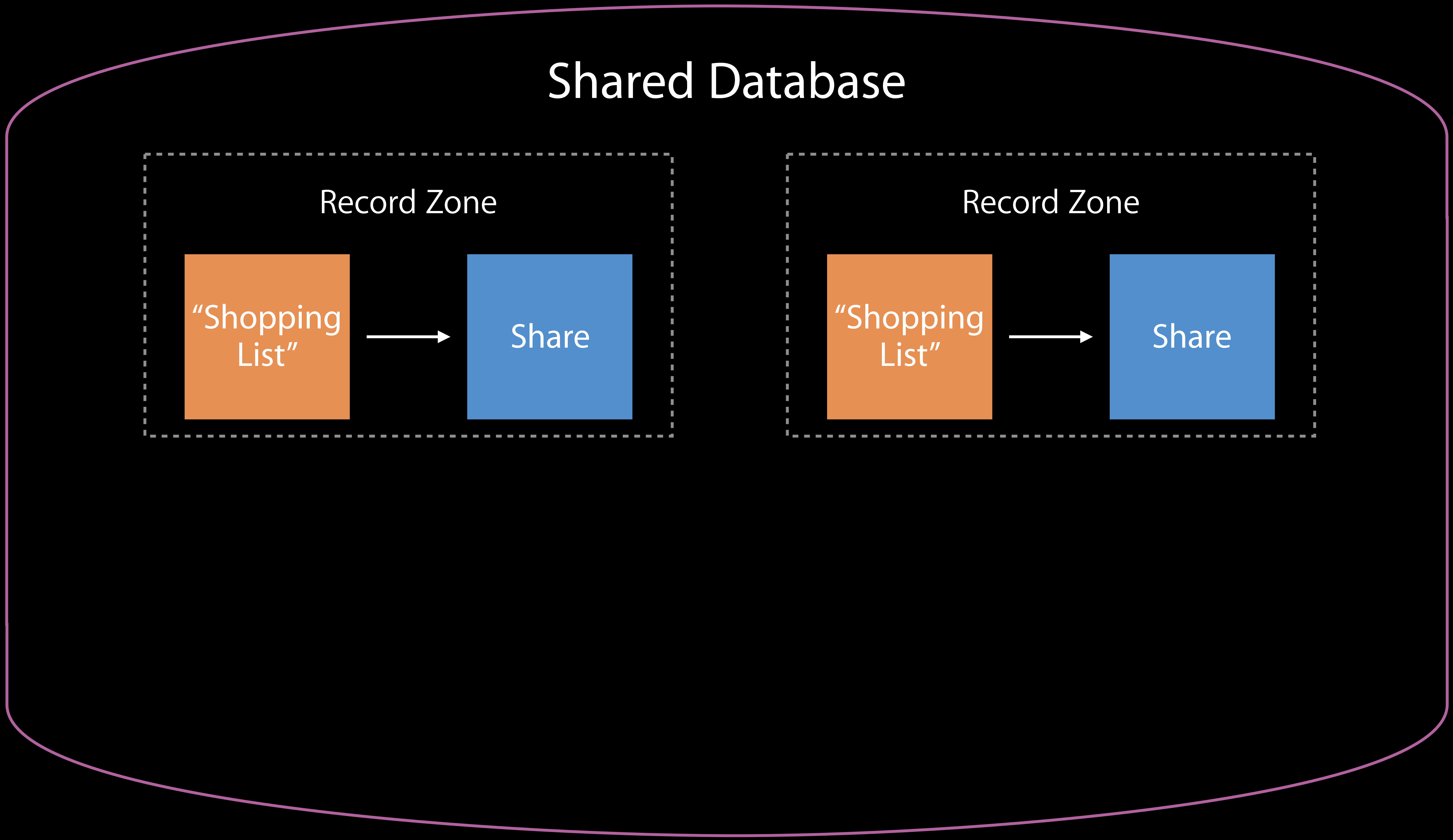
Zones in Shared Database

Two owners, two shares



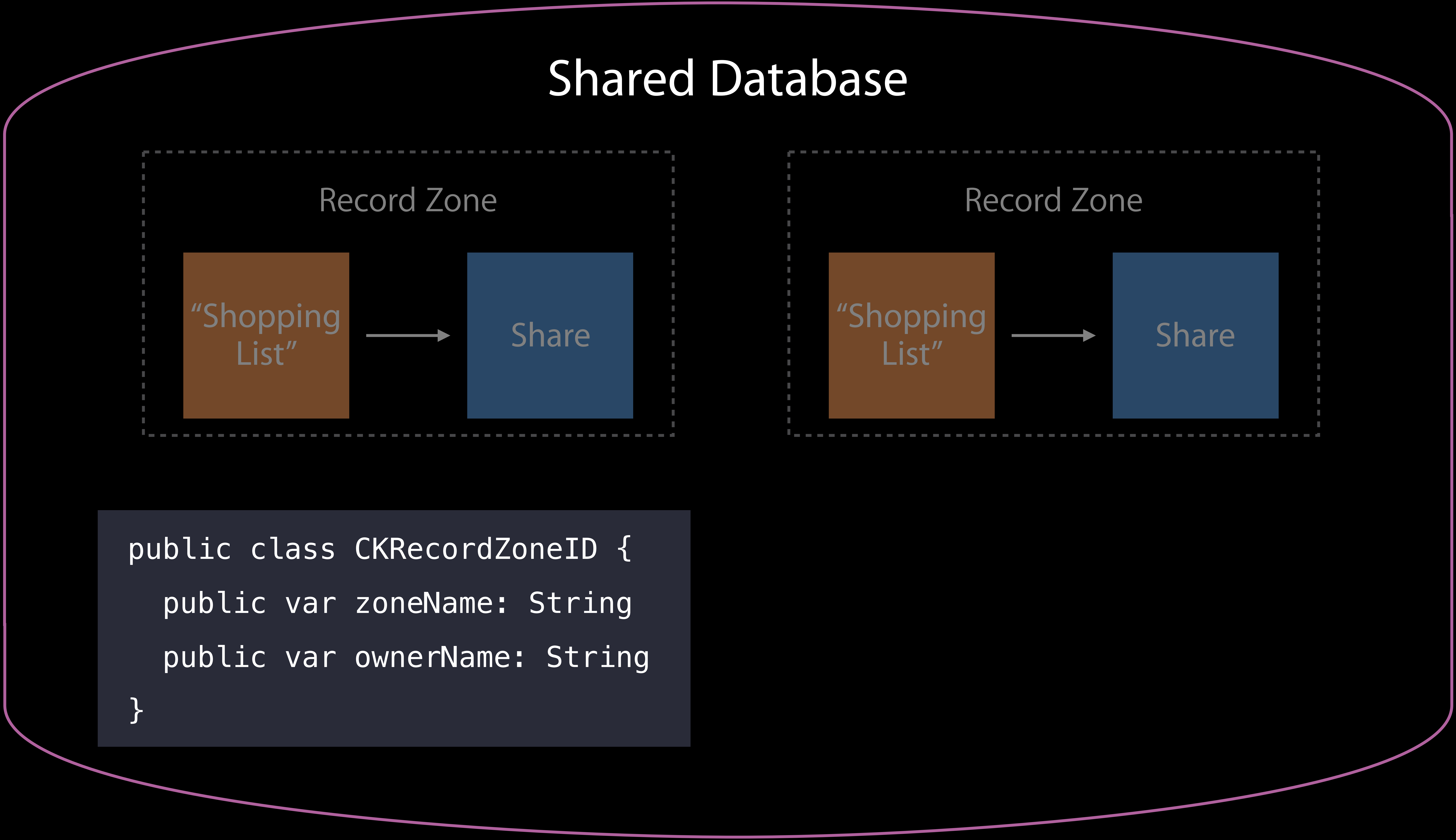
Zones in Shared Database

Two owners, two shares, two zones



Zones in Shared Database

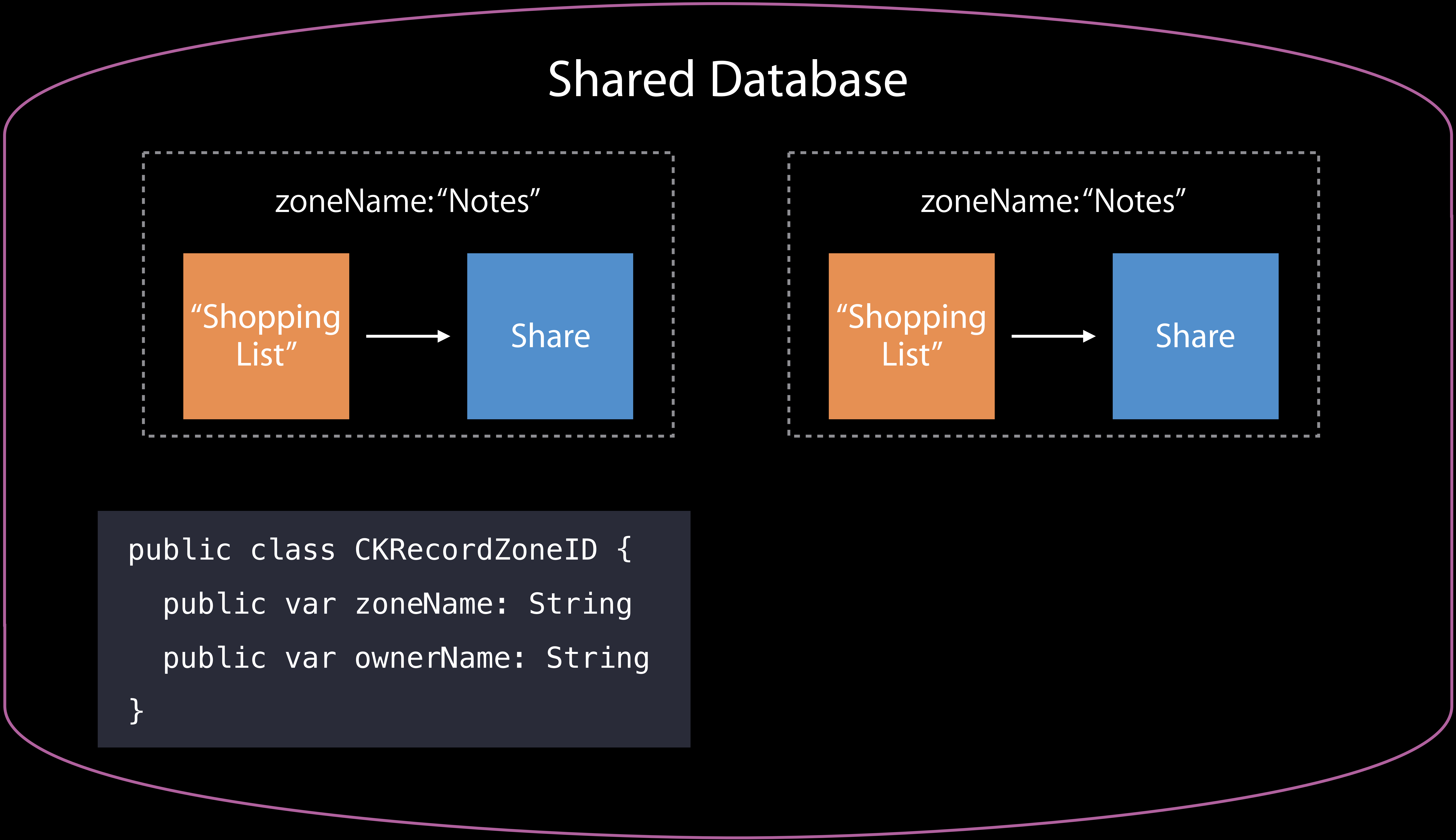
Two owners, two shares, two zones



```
public class CKRecordZoneID {
  public var zoneName: String
  public var ownerName: String
}
```

Zones in Shared Database

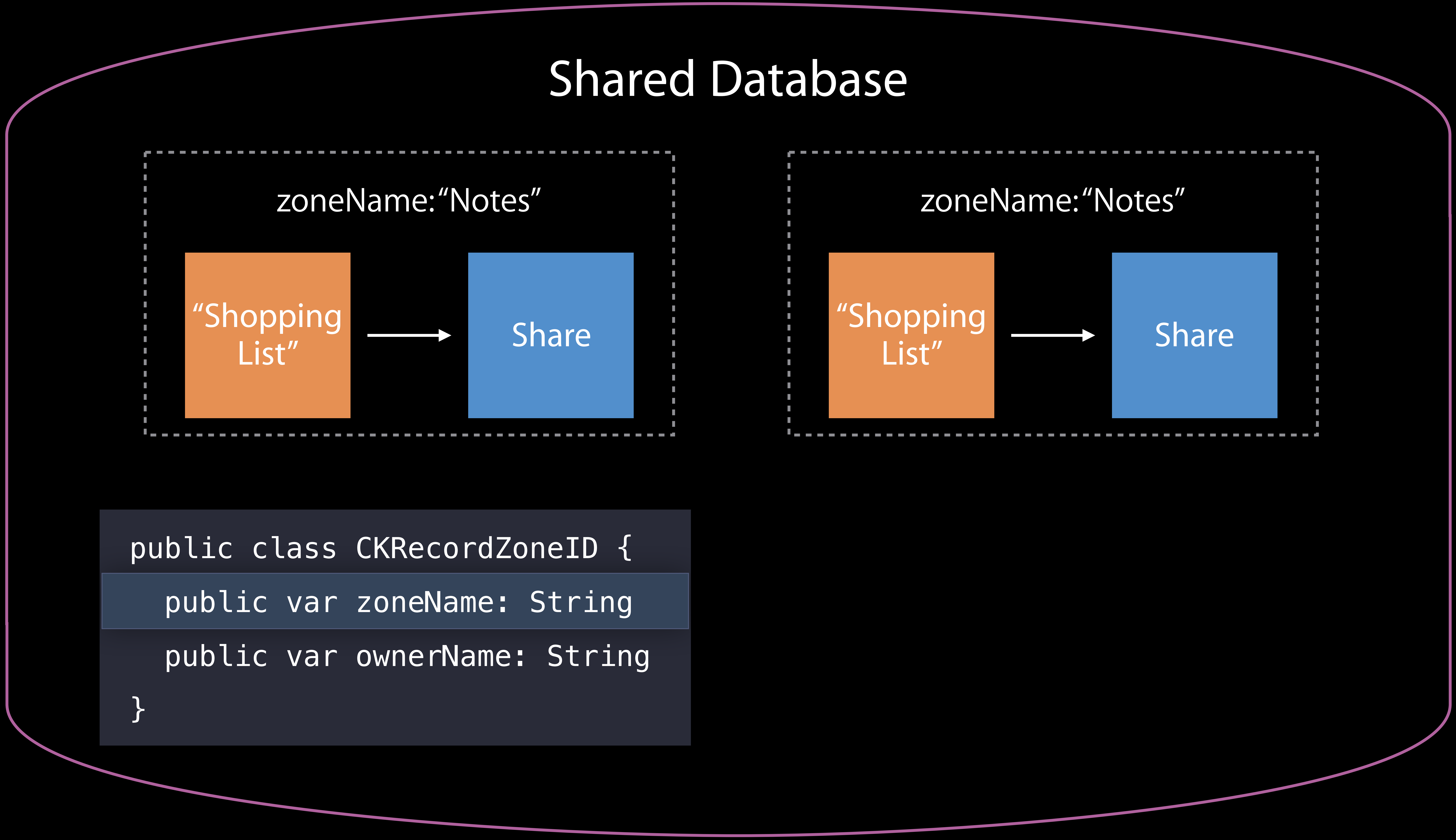
Two owners, two shares, two zones



```
public class CKRecordZoneID {
  public var zoneName: String
  public var ownerName: String
}
```

Zones in Shared Database

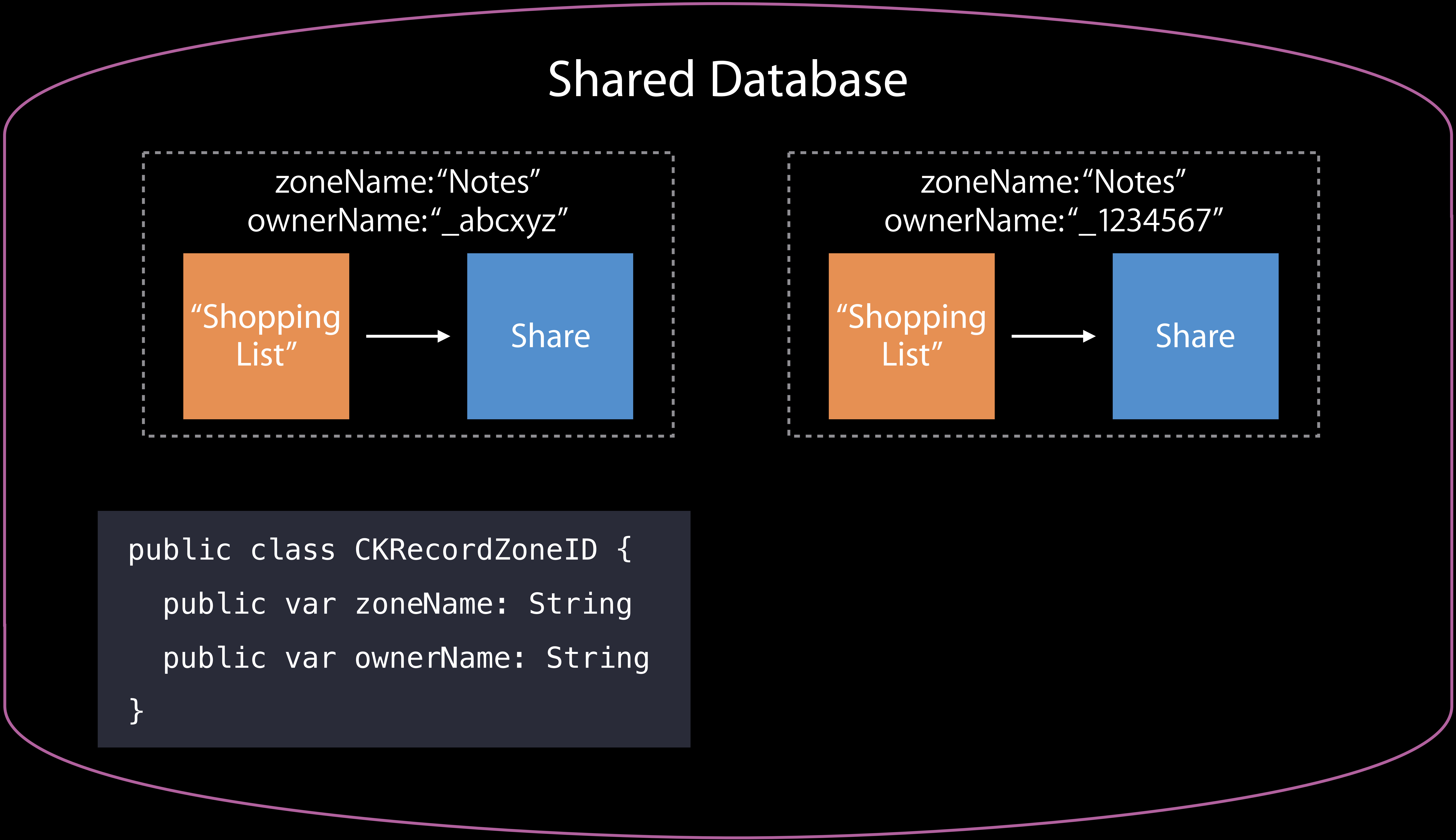
Two owners, two shares, two zones



```
public class CKRecordZoneID {
    public var zoneName: String
    public var ownerName: String
}
```

Zones in Shared Database

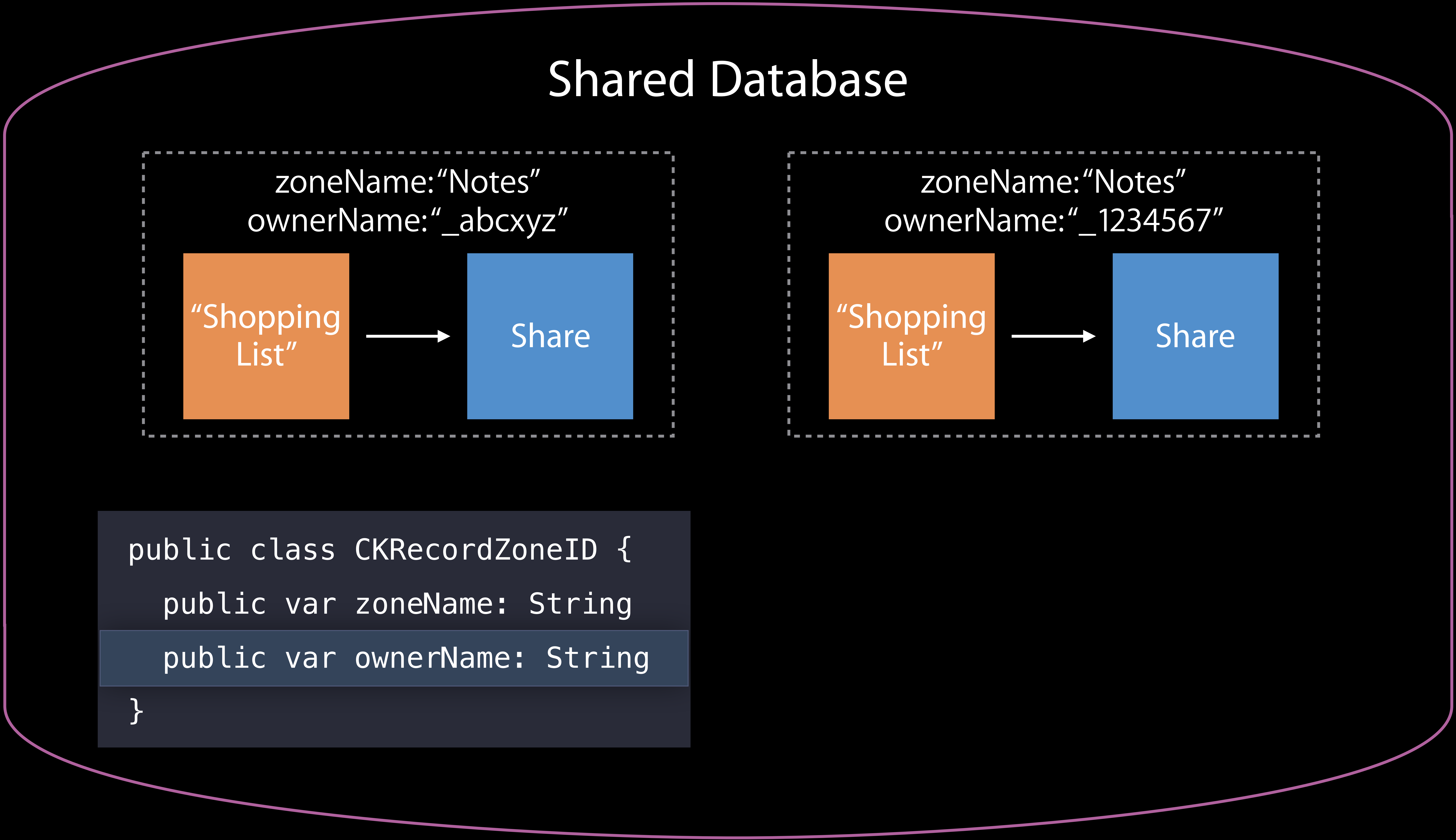
Two owners, two shares, two zones



```
public class CKRecordZoneID {
  public var zoneName: String
  public var ownerName: String
}
```

Zones in Shared Database

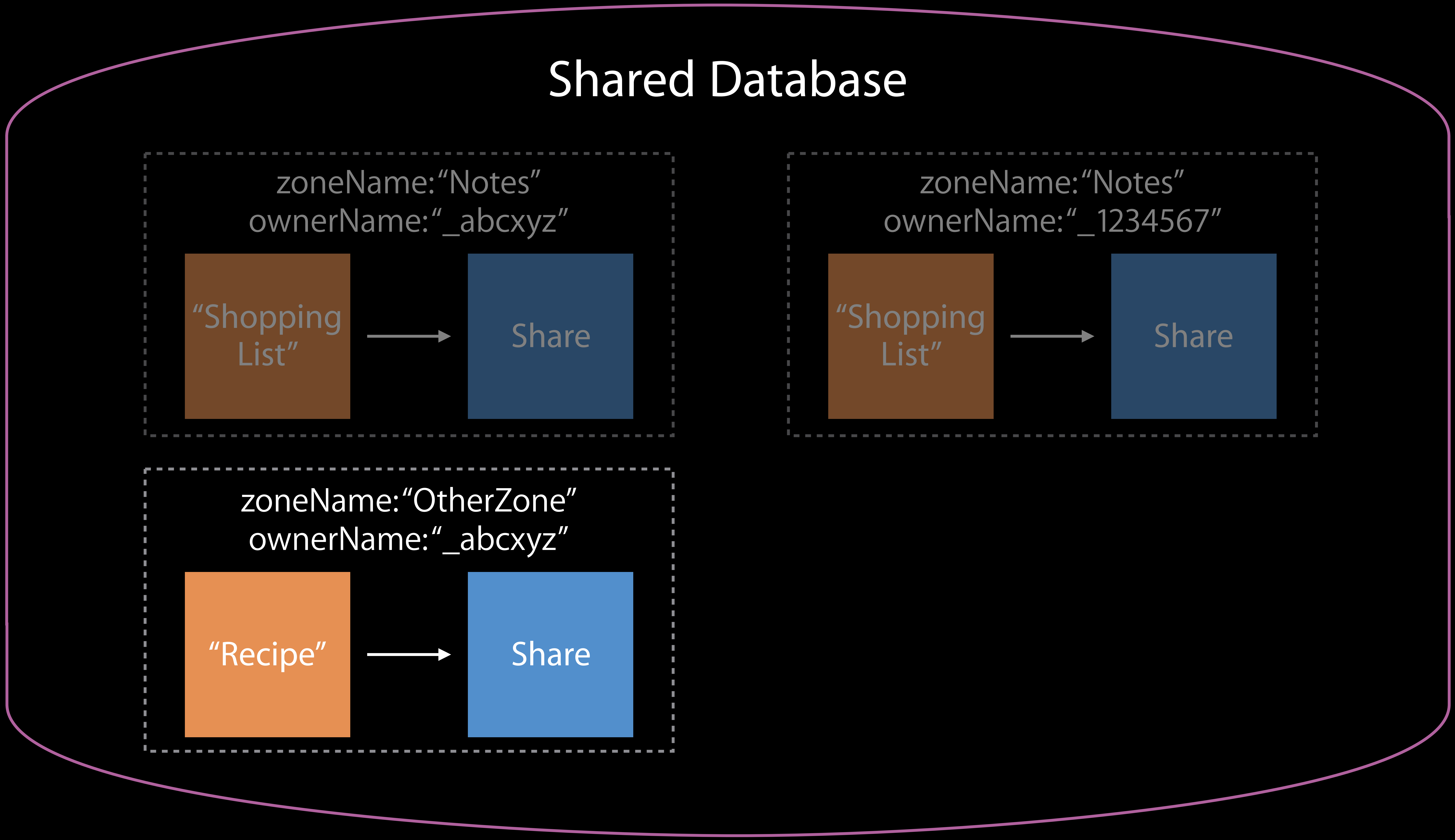
Two owners, two shares, two zones



```
public class CKRecordZoneID {
  public var zoneName: String
  public var ownerName: String
}
```

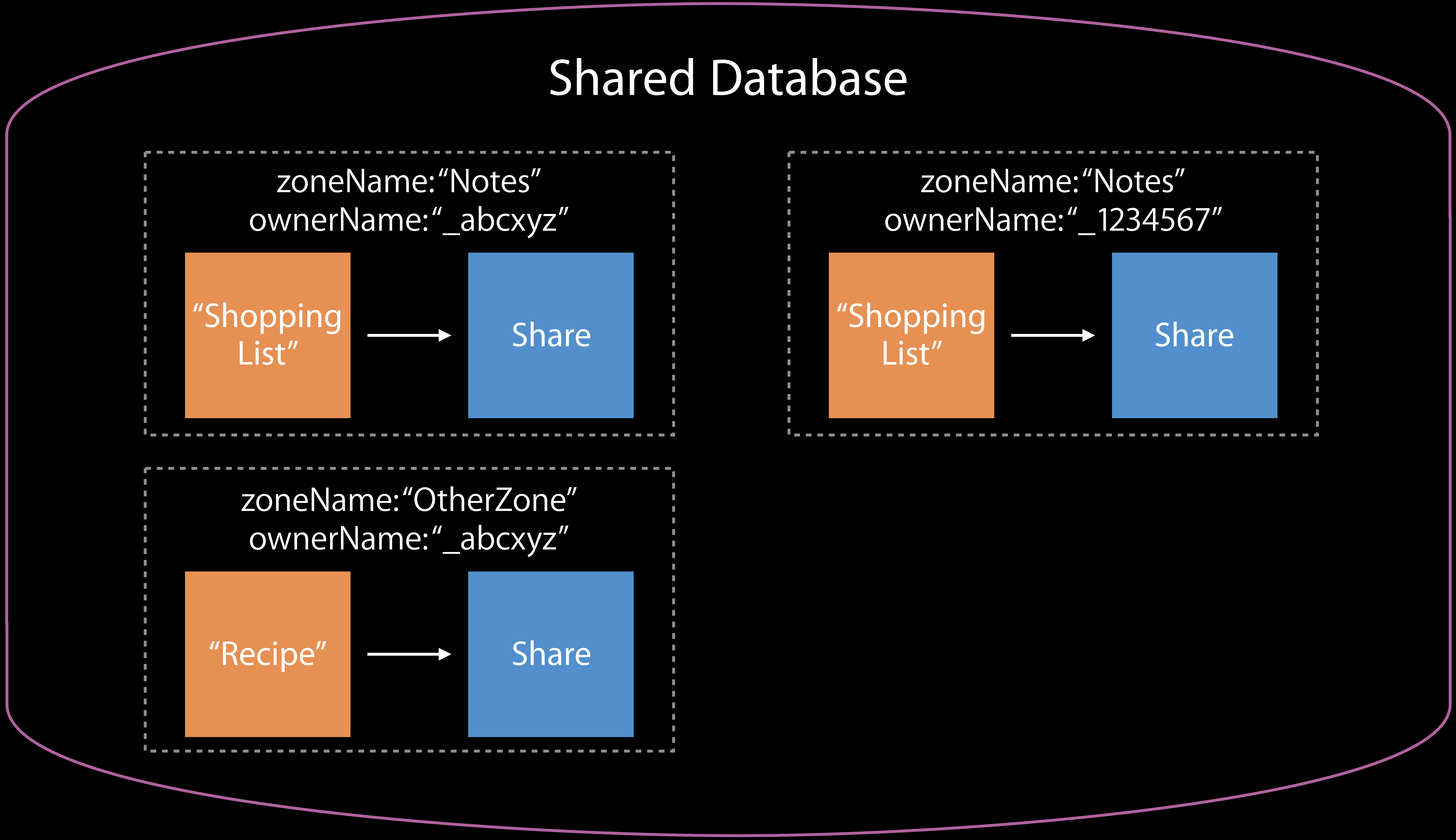
Zones in Shared Database

Two owners, three shares, three zones



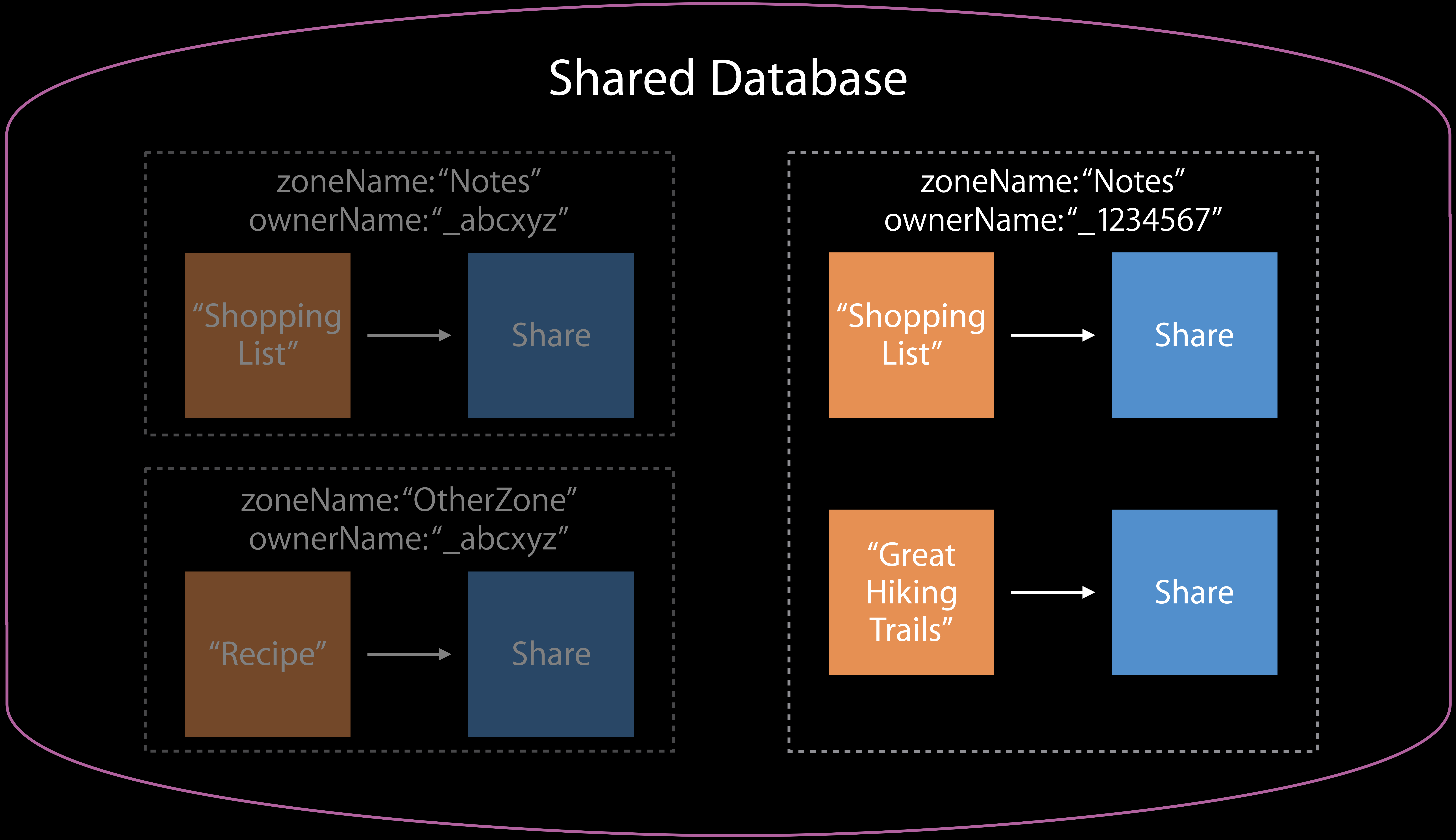
Zones in Shared Database

Two owners, three shares, three zones



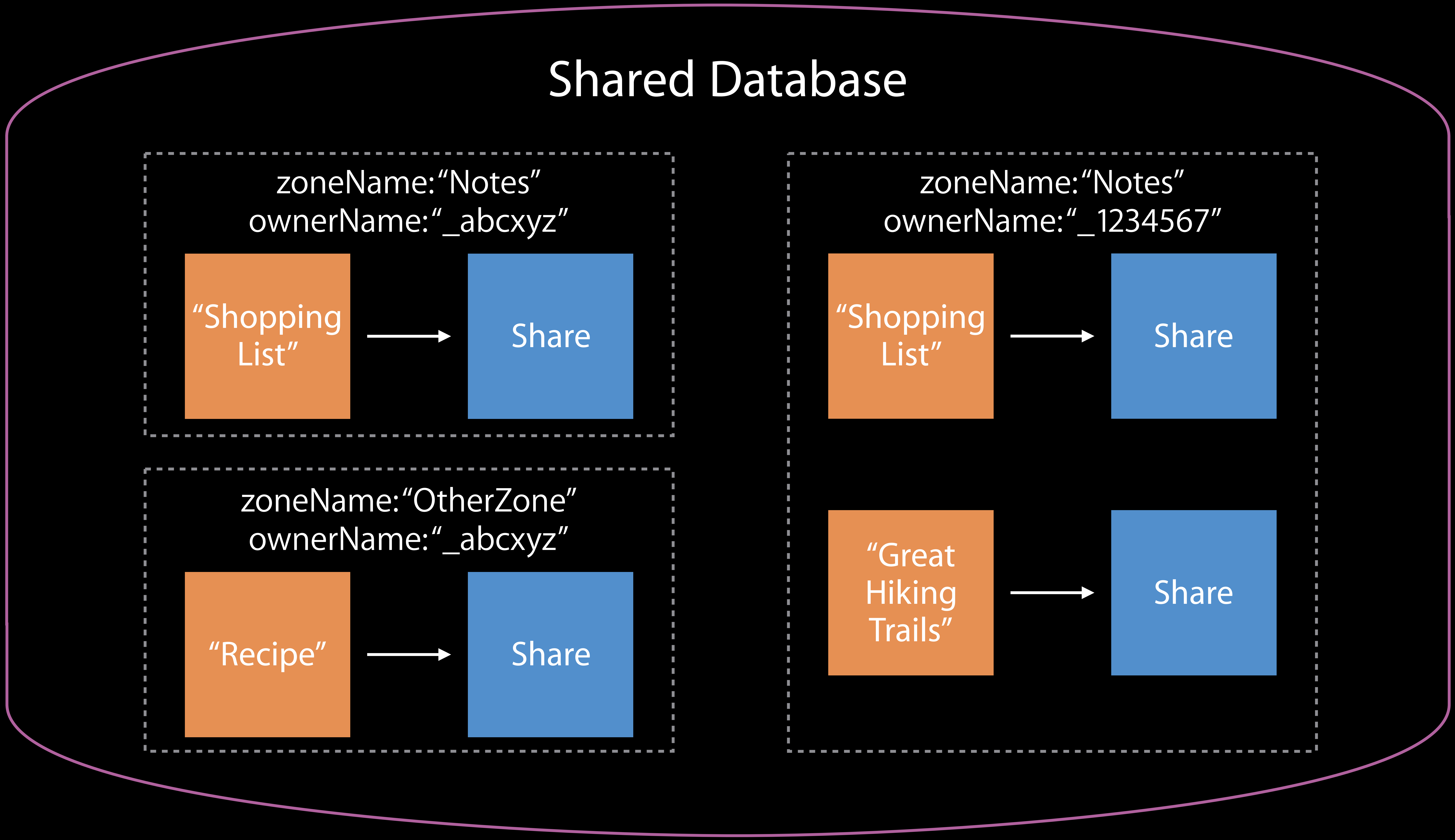
Zones in Shared Database

Two owners, four shares, three zones



Zones in Shared Database

Two owners, four shares, three zones



CKShare

Prerequisite—Owner has existing Record(s) to share

“Shopping List”

CKShare

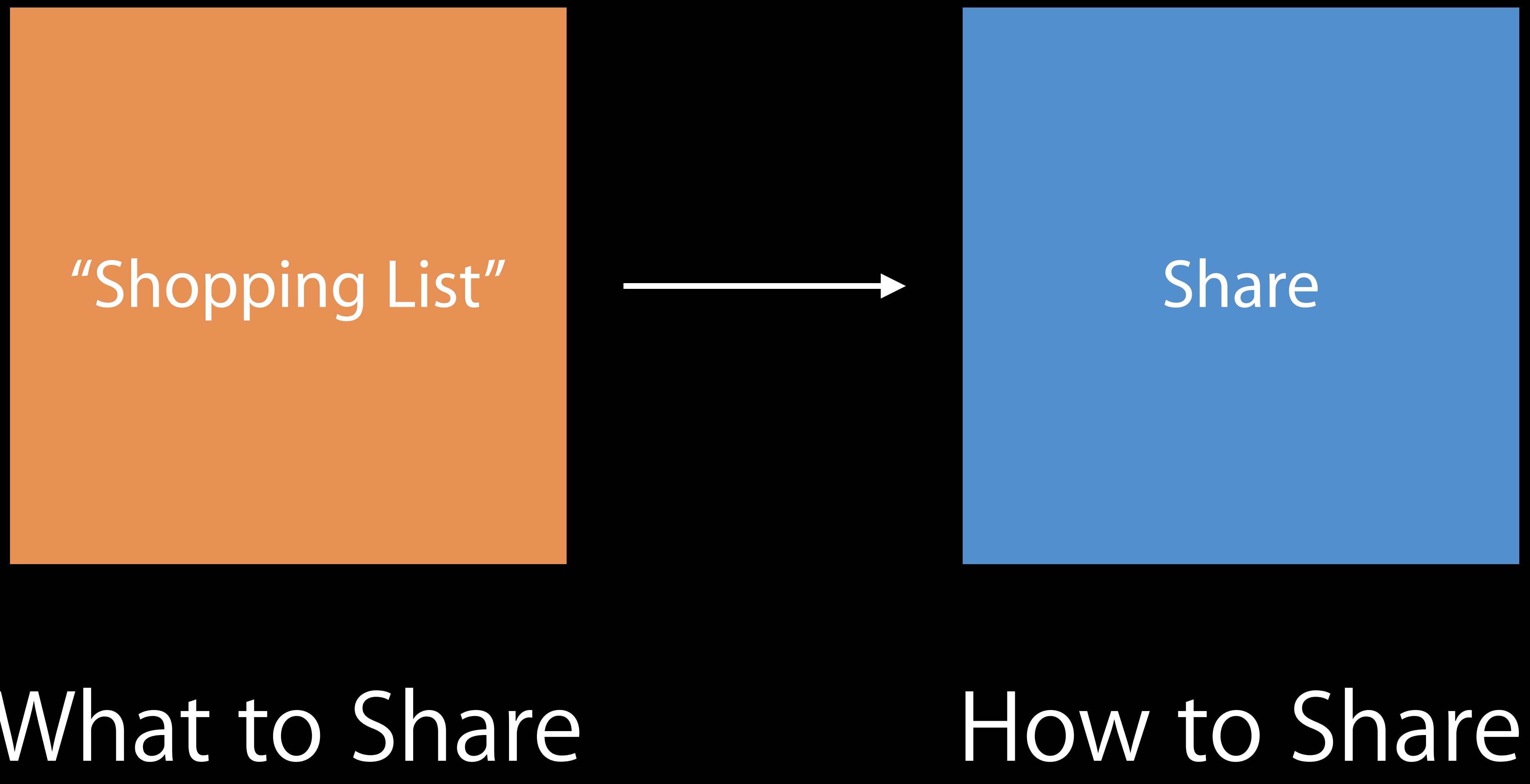
Prerequisite—Owner has existing Record(s) to share

“Shopping List”

What to Share

CKShare

Prerequisite—Owner has existing Record(s) to share



CKShare

Is a CKRecord containing access controls for shared data

Every CKShare has additional properties beyond a basic CKRecord

CKShare

Is a CKRecord containing access controls for shared data

Every CKShare has additional properties beyond a basic CKRecord

```
public class CKShare : CKRecord {
   public var participants: [CKShareParticipant]
}
```

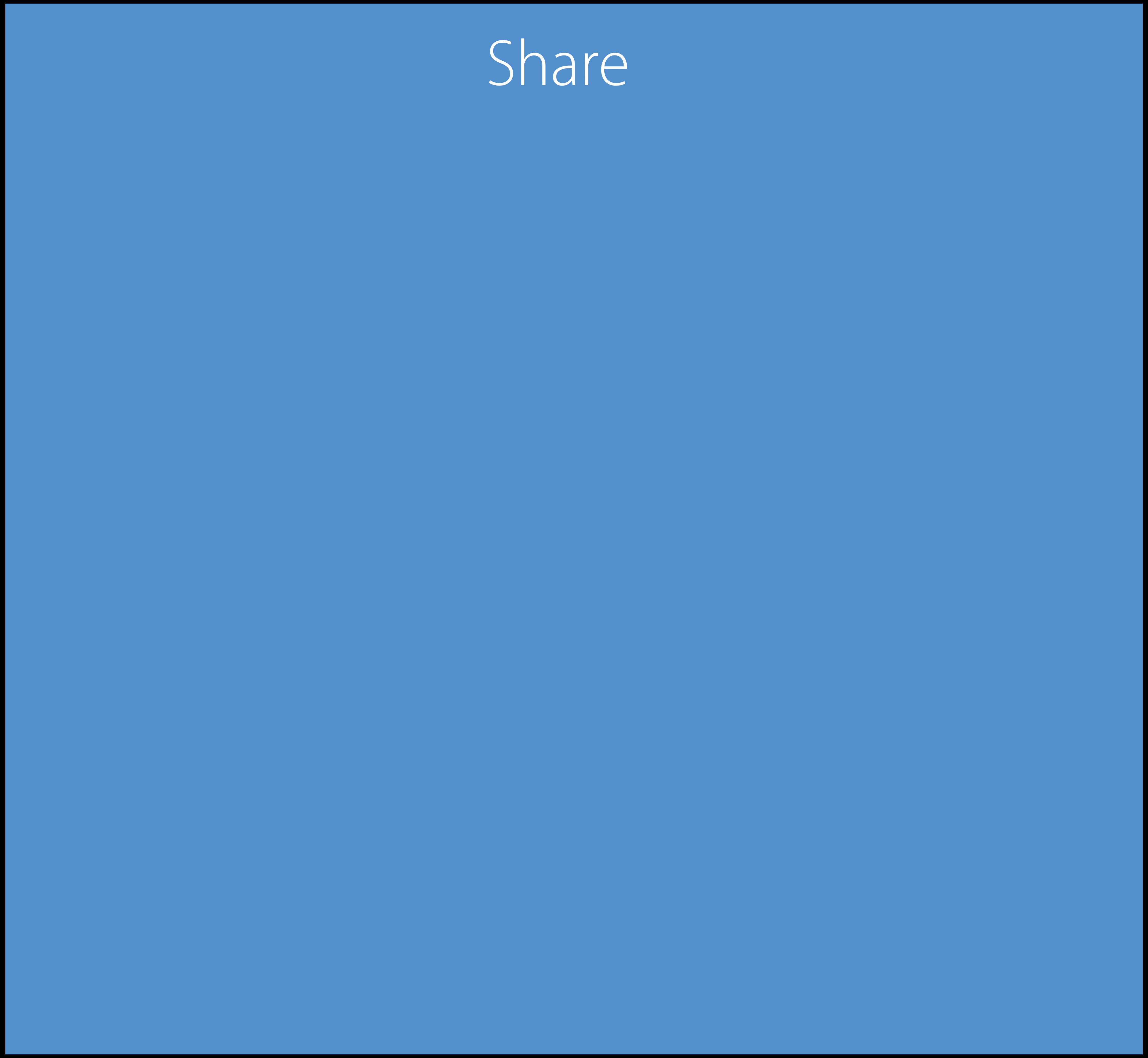
```
public class CKShareParticipant
```

CKShare Lifecycle—Invite-Only

Owner sets up the Share

CKShare Lifecycle—Invite-Only

Owner sets up the Share



1. Create a Share

CKShare Lifecycle—Invite-Only

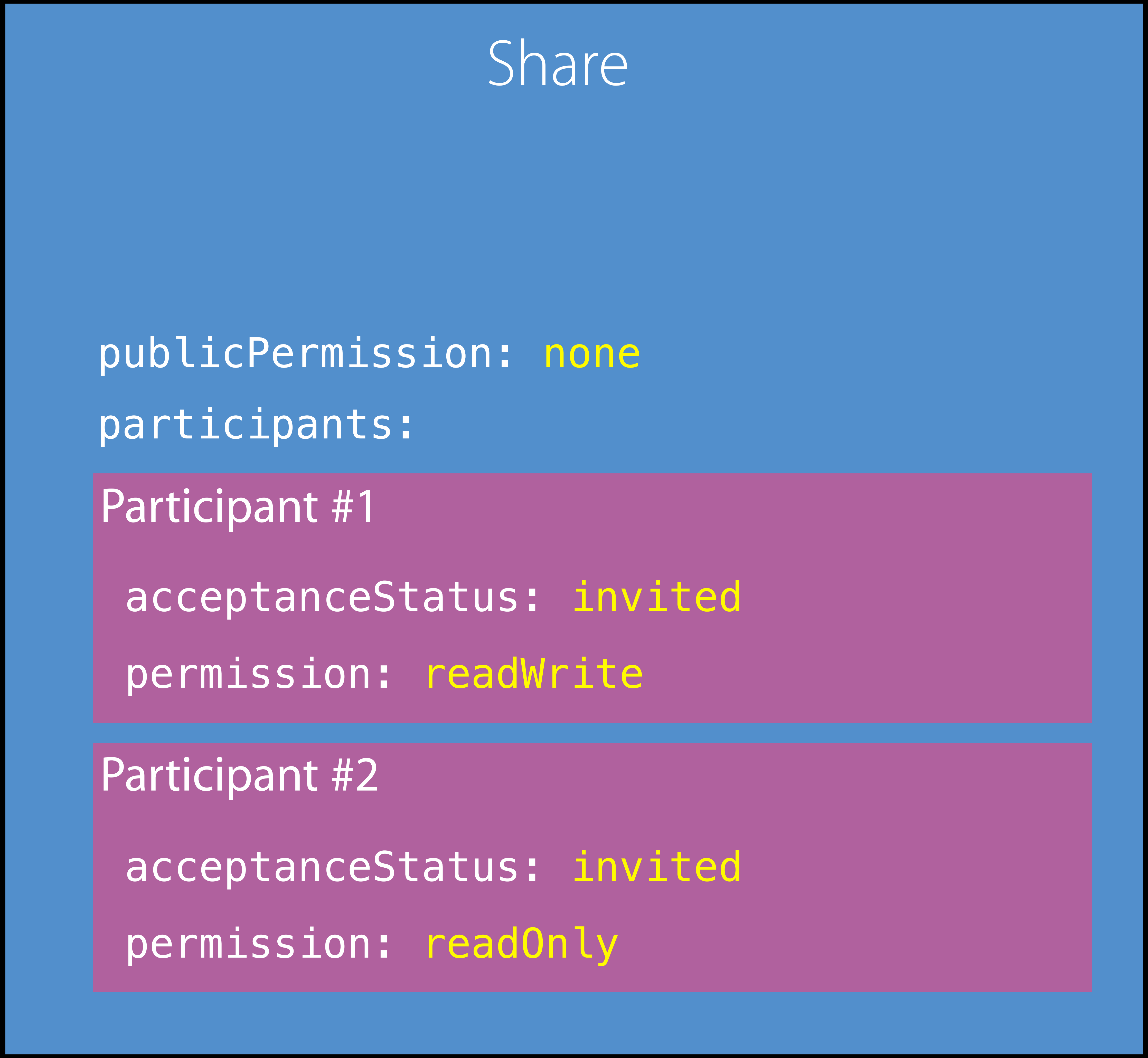
Owner sets up the Share

Share
publicPermission: none

1. Create a Share
2. publicPermission=none

CKShare Lifecycle—Invite-Only

Owner sets up the Share



Share

```
publicPermission: none
participants:
```

Participant #1

```
 acceptanceStatus: invited
 permission: readWrite
```

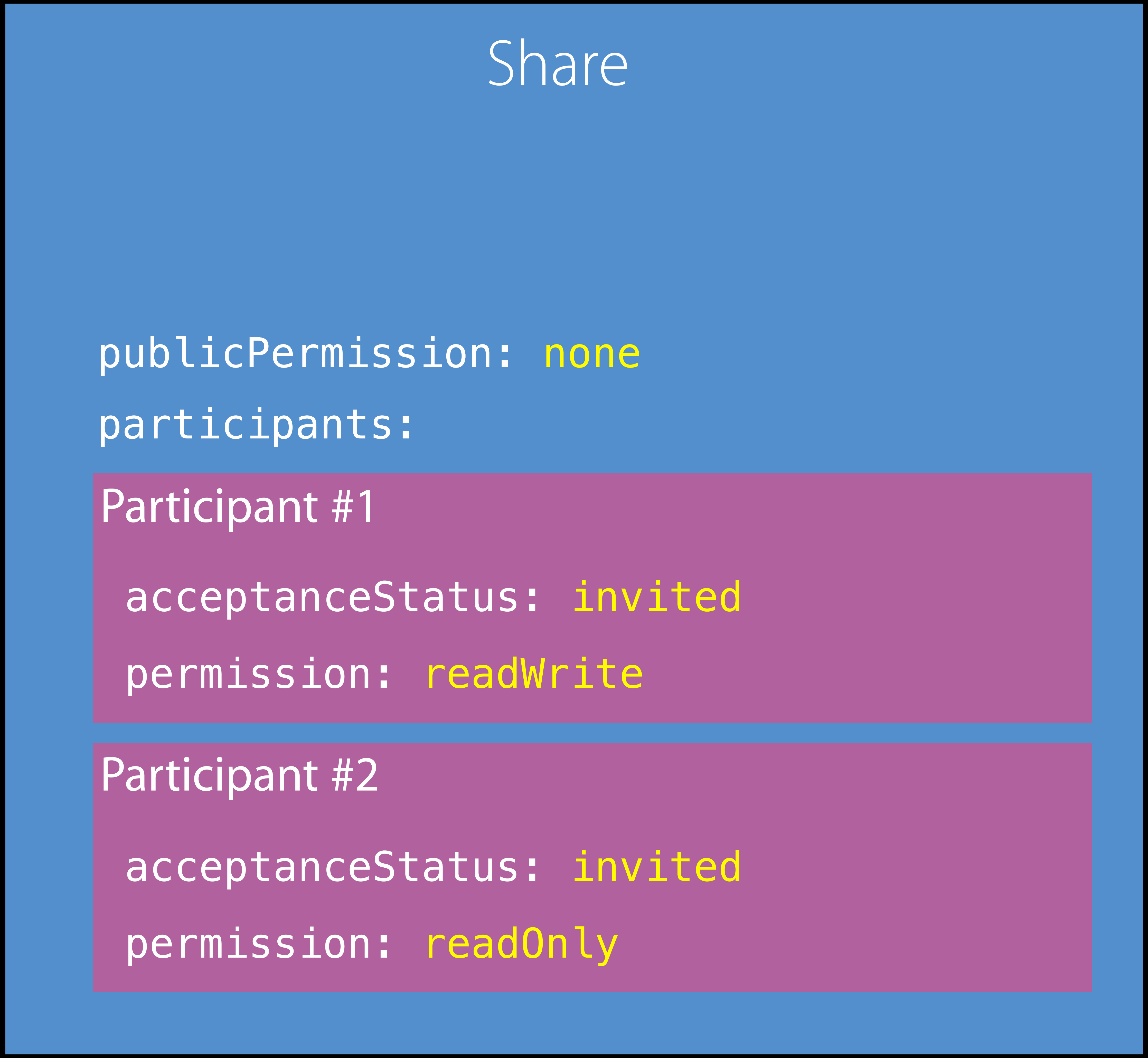
Participant #2

```
 acceptanceStatus: invited
 permission: readOnly
```

1. Create a Share
2. publicPermission=none
3. Add Participant
    1. acceptanceStatus=invited
    2. Owner determines each participant's permission

CKShare Lifecycle—Invite-Only

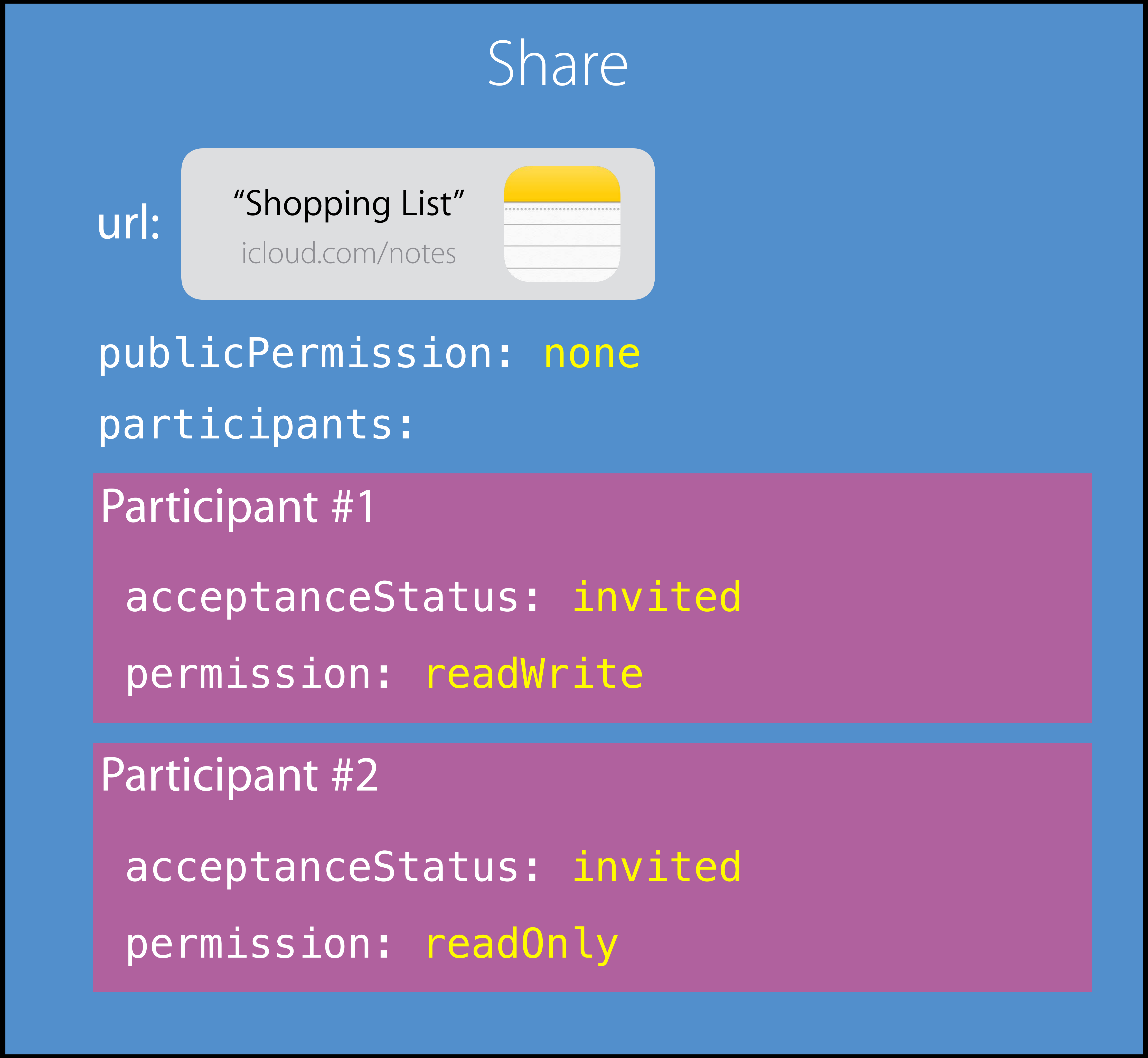
Owner sets up the Share



1. Create a Share
2. publicPermission=none
3. Add Participant
    1. acceptanceStatus=invited
    2. Owner determines each participant's permission
4. Save the Share

CKShare Lifecycle—Invite-Only

Owner sets up the Share



```
publicPermission: none
participants:
```

Participant #1

```
acceptanceStatus: invited
permission: readWrite
```

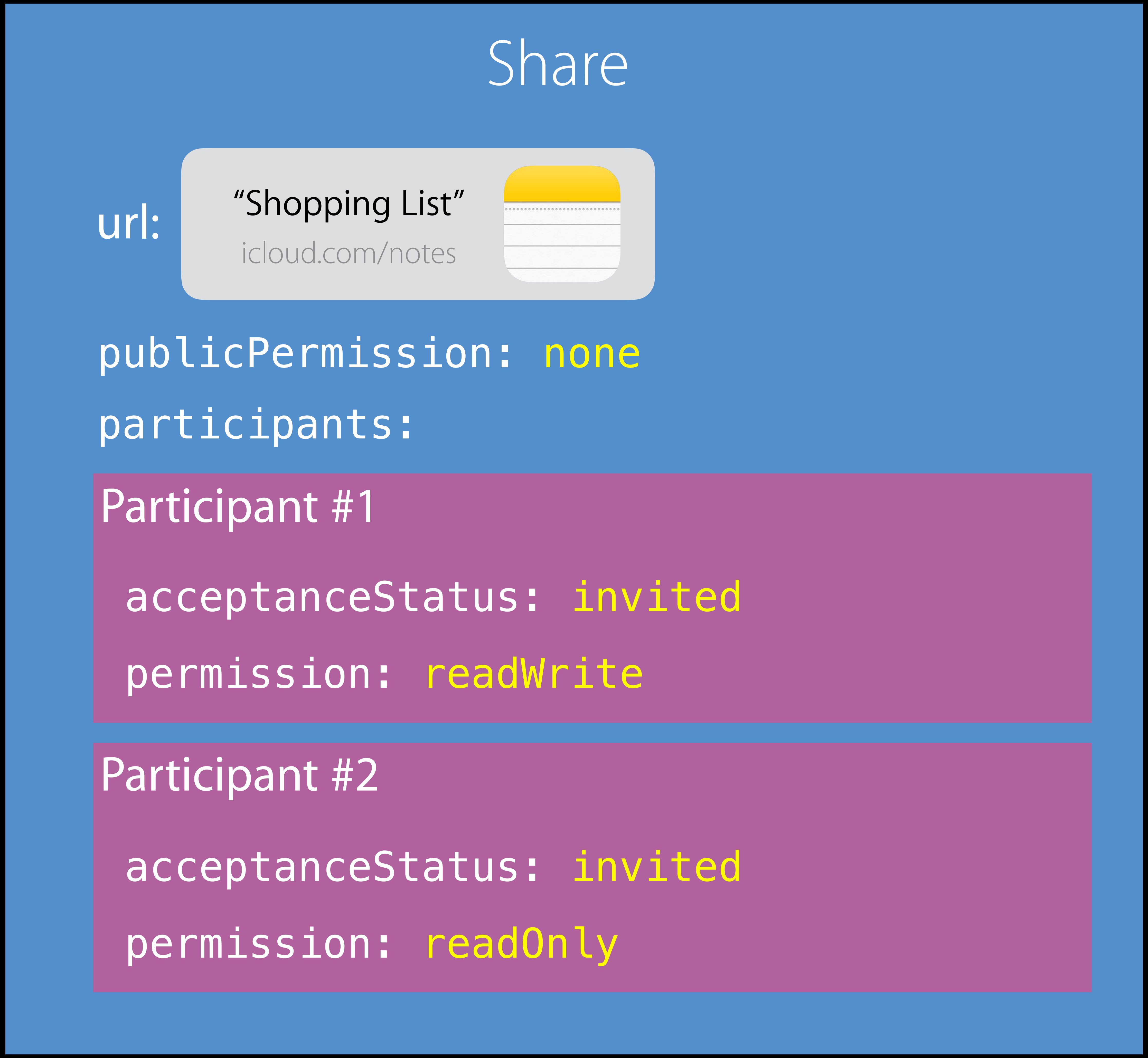
Participant #2

```
acceptanceStatus: invited
permission: readOnly
```

1. Create a Share
2. publicPermission=none
3. Add Participant
    1. acceptanceStatus=invited
    2. Owner determines each participant’s permission
4. Save the Share
5. Owner gets URL

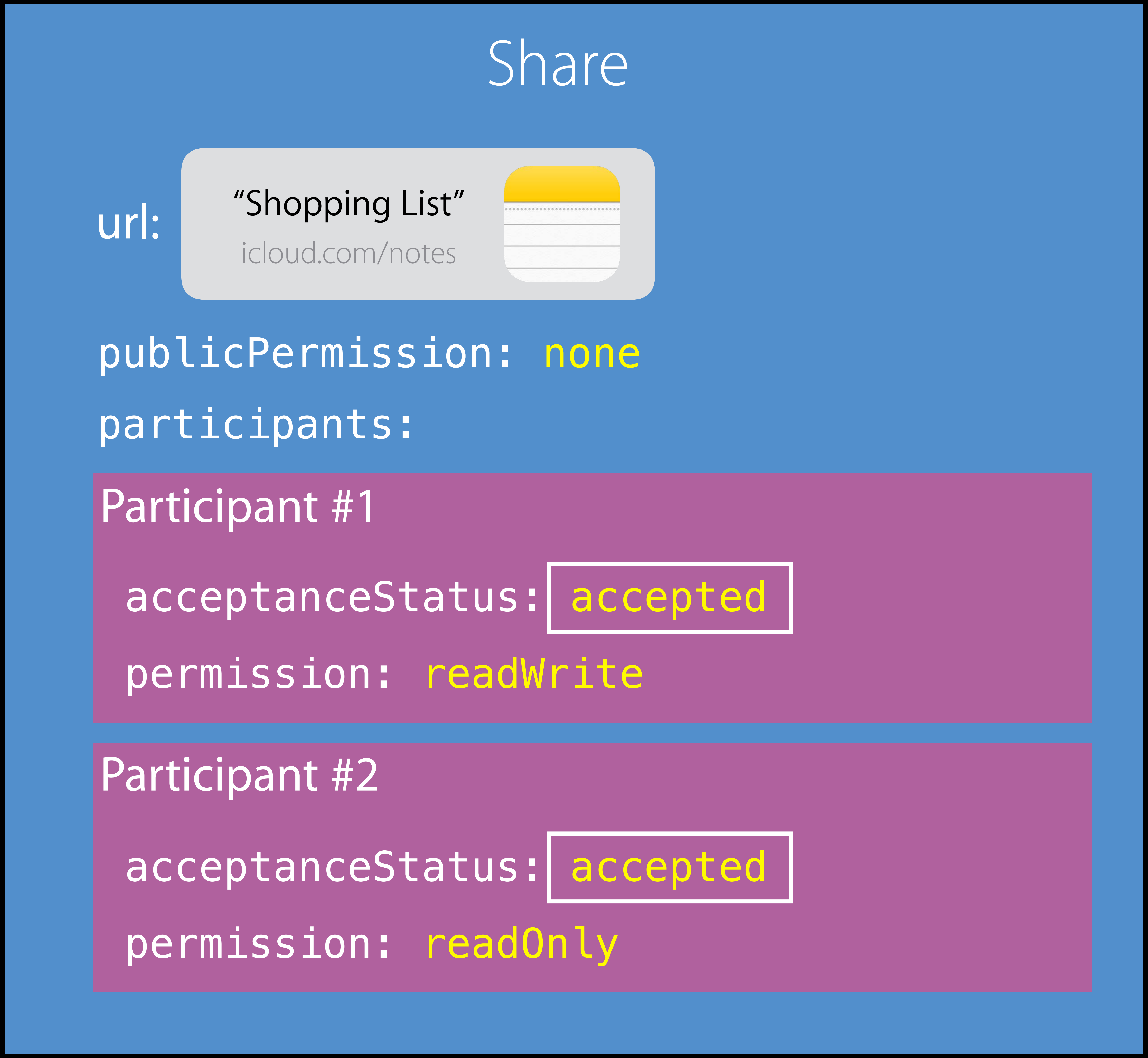
CKShare Lifecycle—Invite-Only

Participant joins the Share



CKShare Lifecycle—Invite-Only

Participant joins the Share



Share

url: "Shopping List" icloud.com/notes

```
publicPermission: none
participants:
```

Participant #1

```
acceptanceStatus: accepted
permission: readWrite
```

Participant #2

```
acceptanceStatus: accepted
permission: readOnly
```

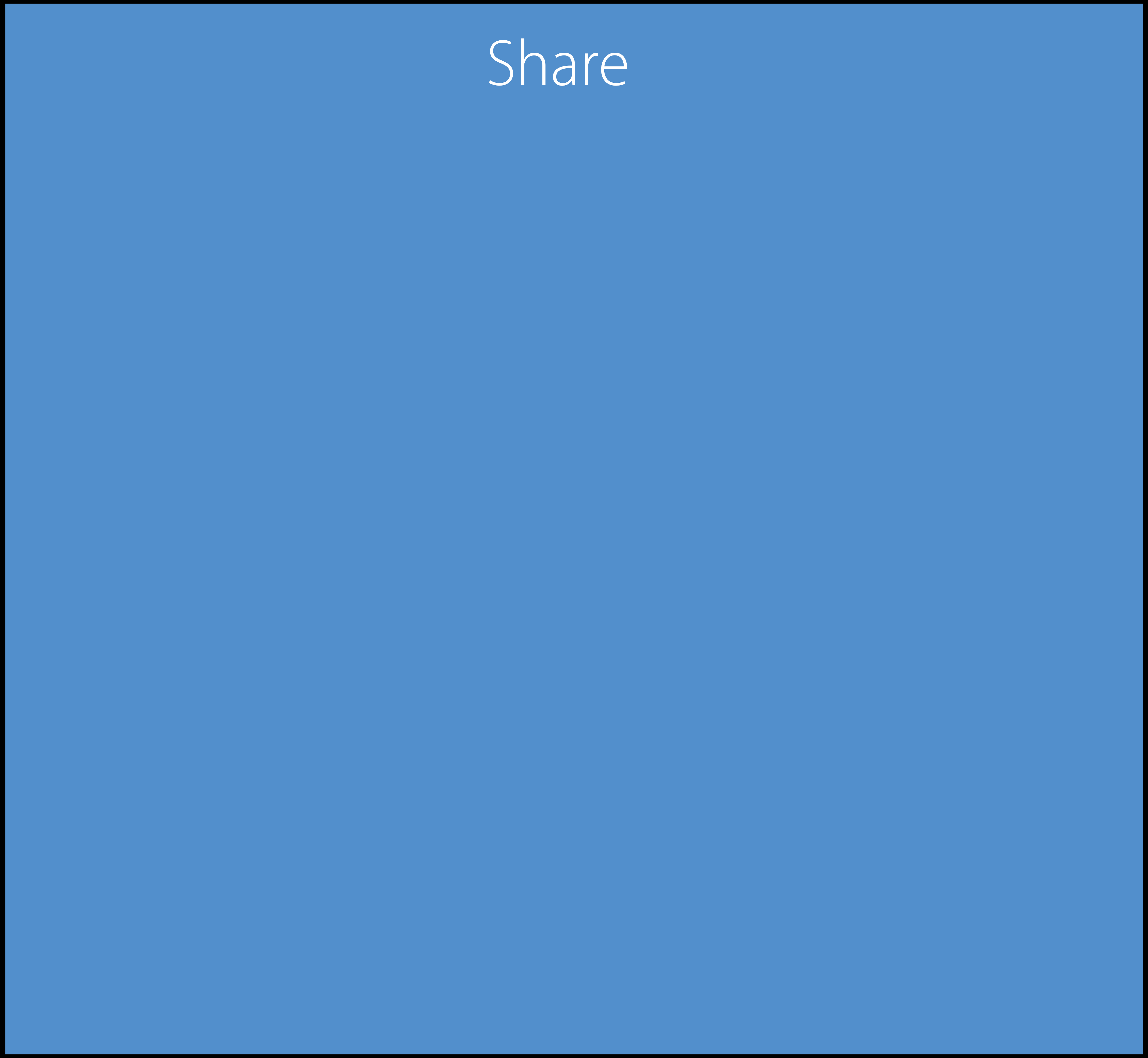
1. Participants accept via URL
   acceptanceStatus=accepted

CKShare Lifecycle—readOnly or readWrite

Owner sets up the Share

CKShare Lifecycle—readOnly or readWrite

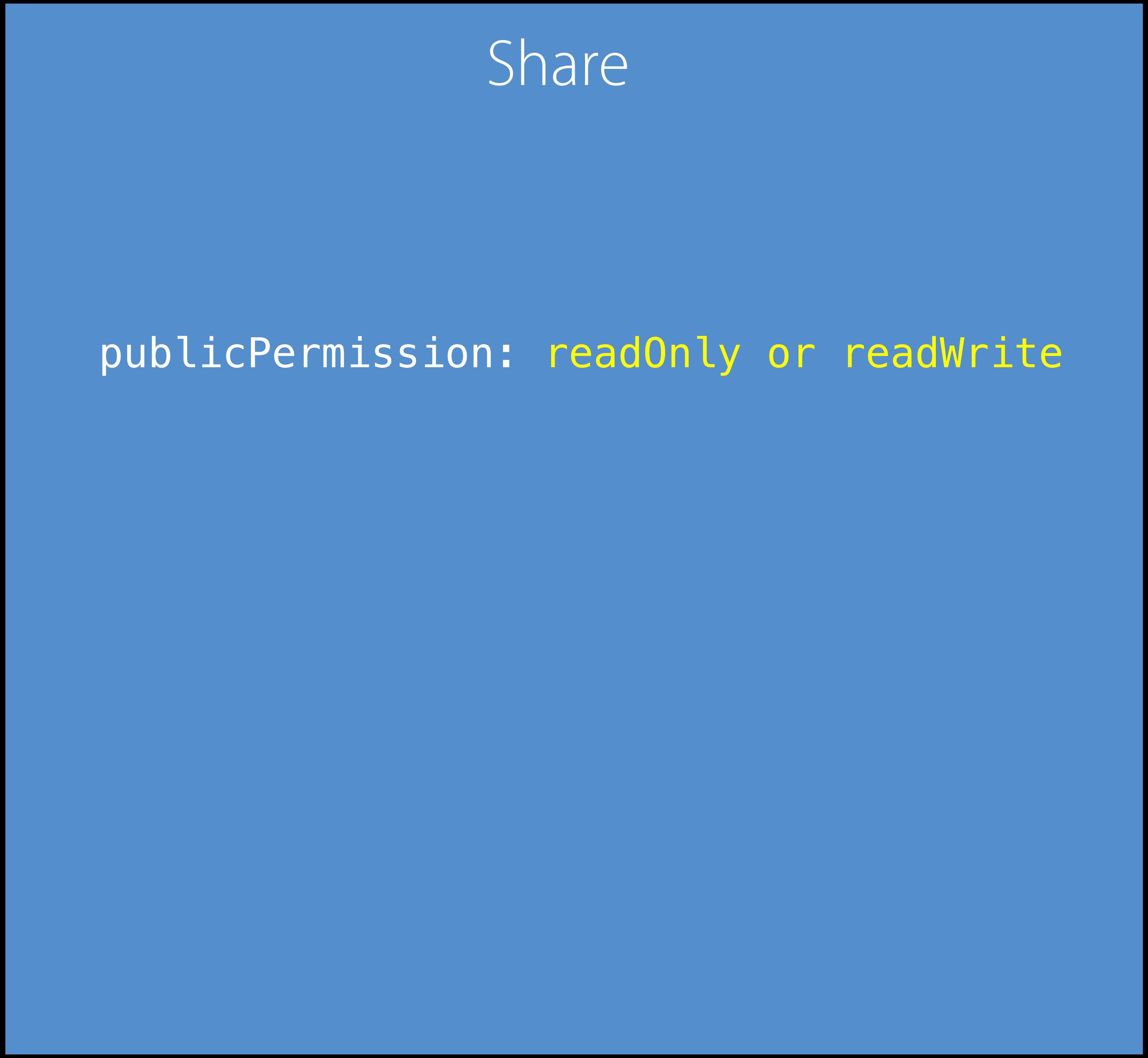
Owner sets up the Share



1. Create a Share

CKShare Lifecycle—readOnly or readWrite

Owner sets up the Share



```
publicPermission: readOnly or readWrite
```

1. Create a Share
2. publicPermission=readOnly or readWrite

CKShare Lifecycle—readOnly or readWrite

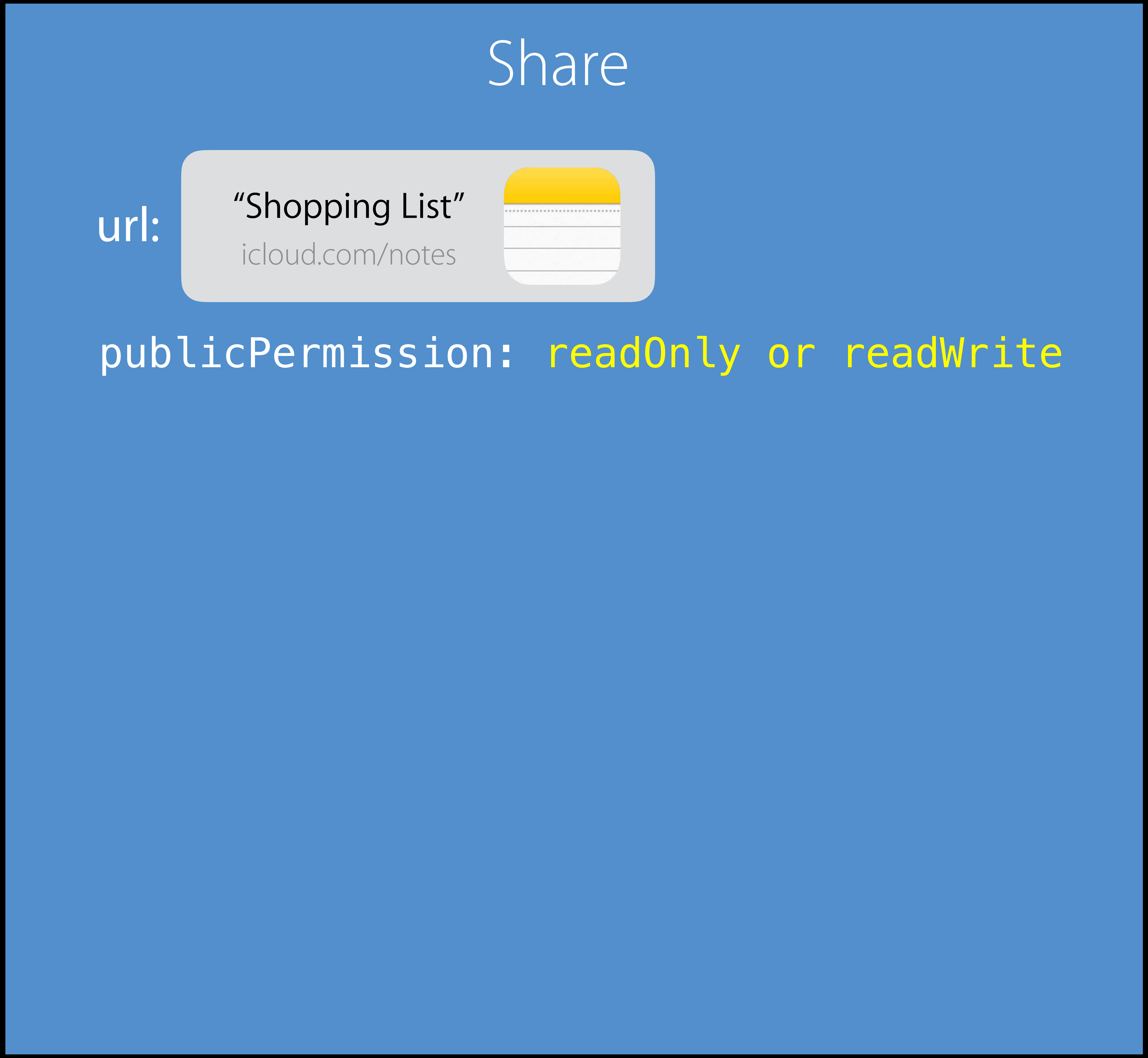
Owner sets up the Share

Share
publicPermission: readOnly or readWrite

1. Create a Share
2. publicPermission=readOnly or readWrite
3. Save the Share

CKShare Lifecycle—readOnly or readWrite

Owner sets up the Share

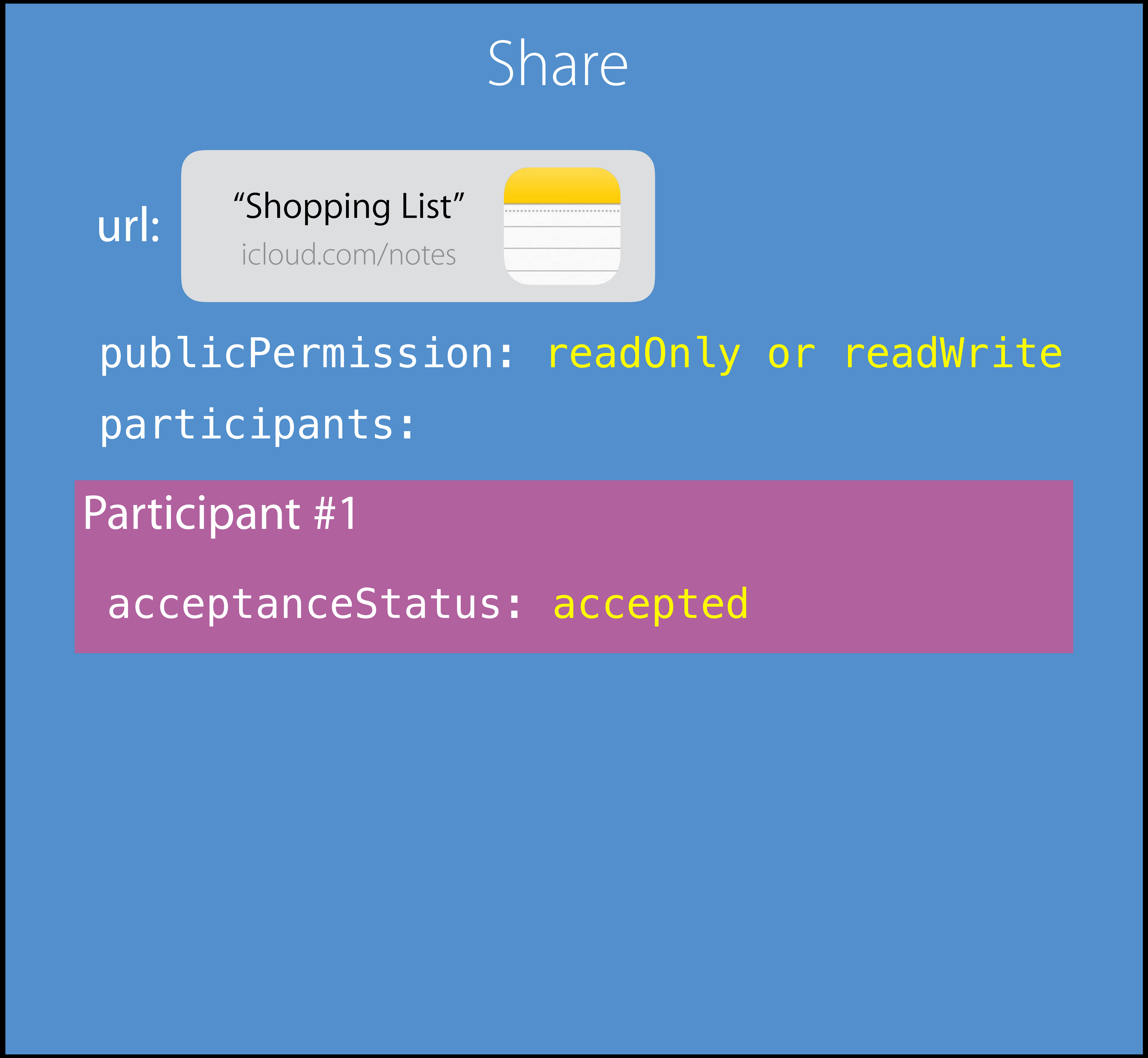


```
publicPermission: readOnly or readWrite
```

1. Create a Share
2. publicPermission=readOnly or readWrite
3. Save the Share
4. Owner gets URL

CKShare Lifecycle—readOnly or readWrite

Participant joins the Share



Share

url: "Shopping List" icloud.com/notes

publicPermission: readOnly or readWrite

participants:

Participant #1

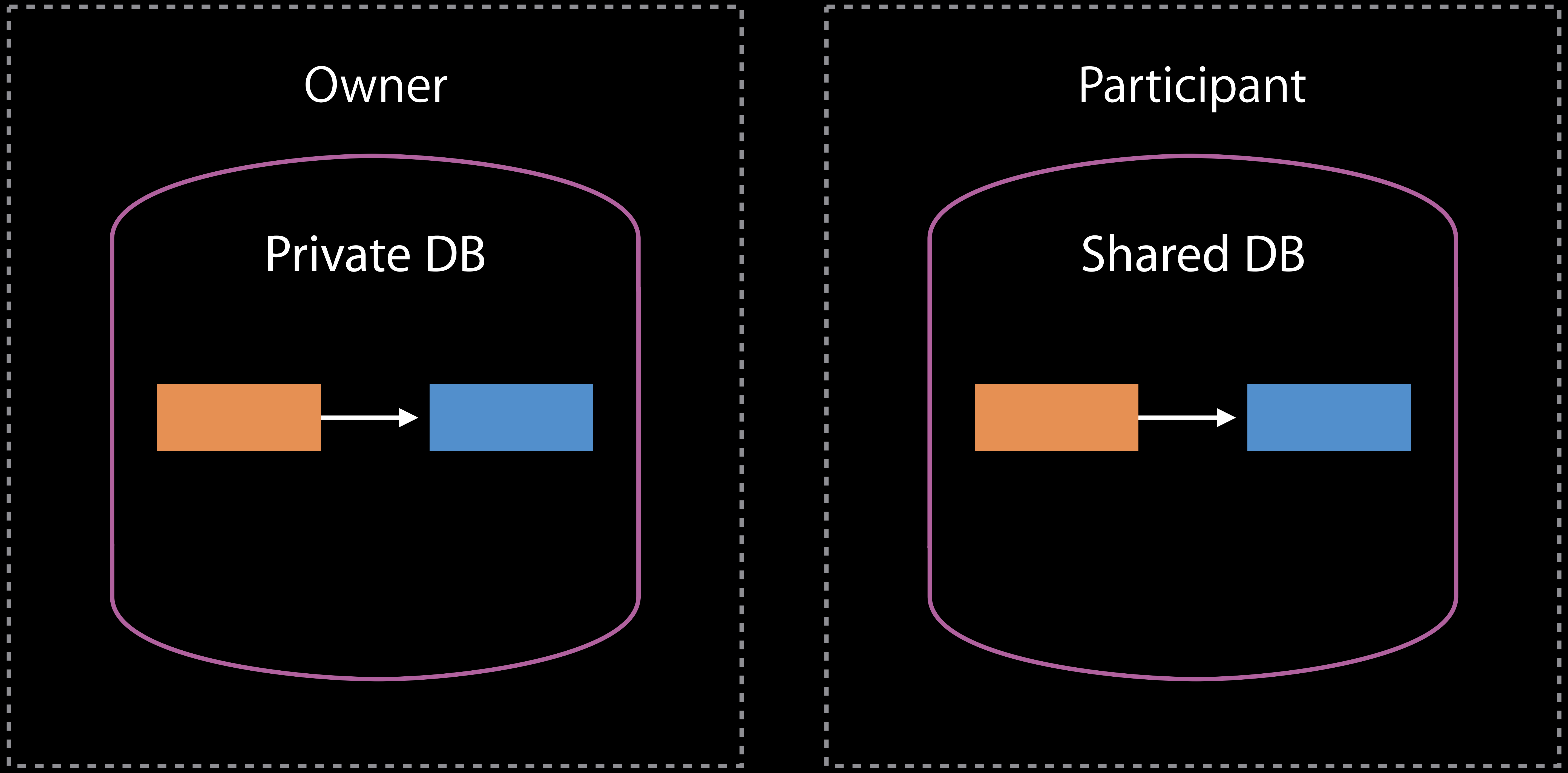
acceptanceStatus: accepted

1. Anyone can join via URL. acceptanceStatus=accepted, permission is the same as the publicPermission

CKShare

Lifecycle—end the share for a participant

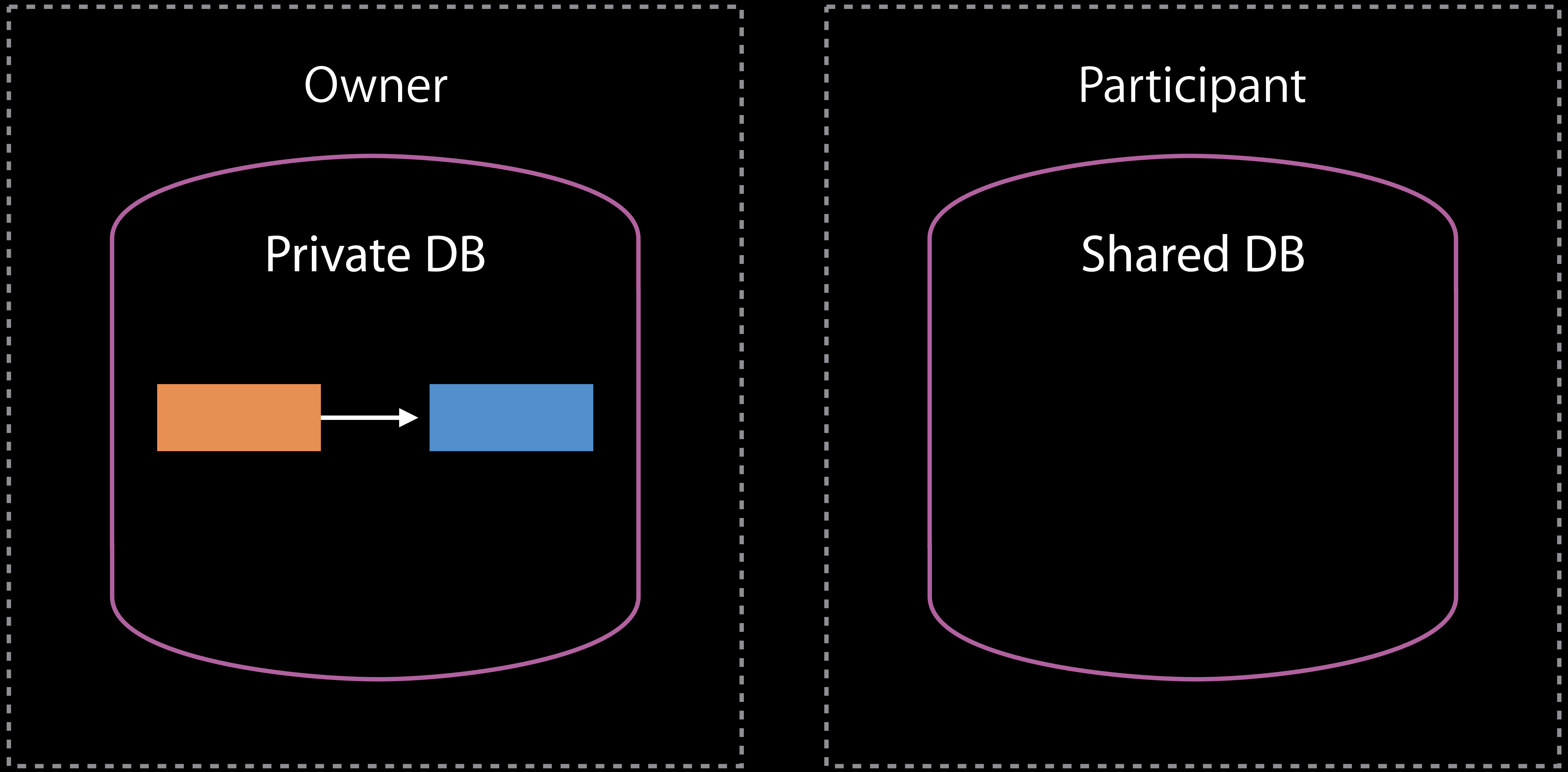
Participant leaves the Share by deleting the Share from their Shared DB



CKShare

Lifecycle—end the share for a participant

Participant leaves the Share by deleting the Share from their Shared DB

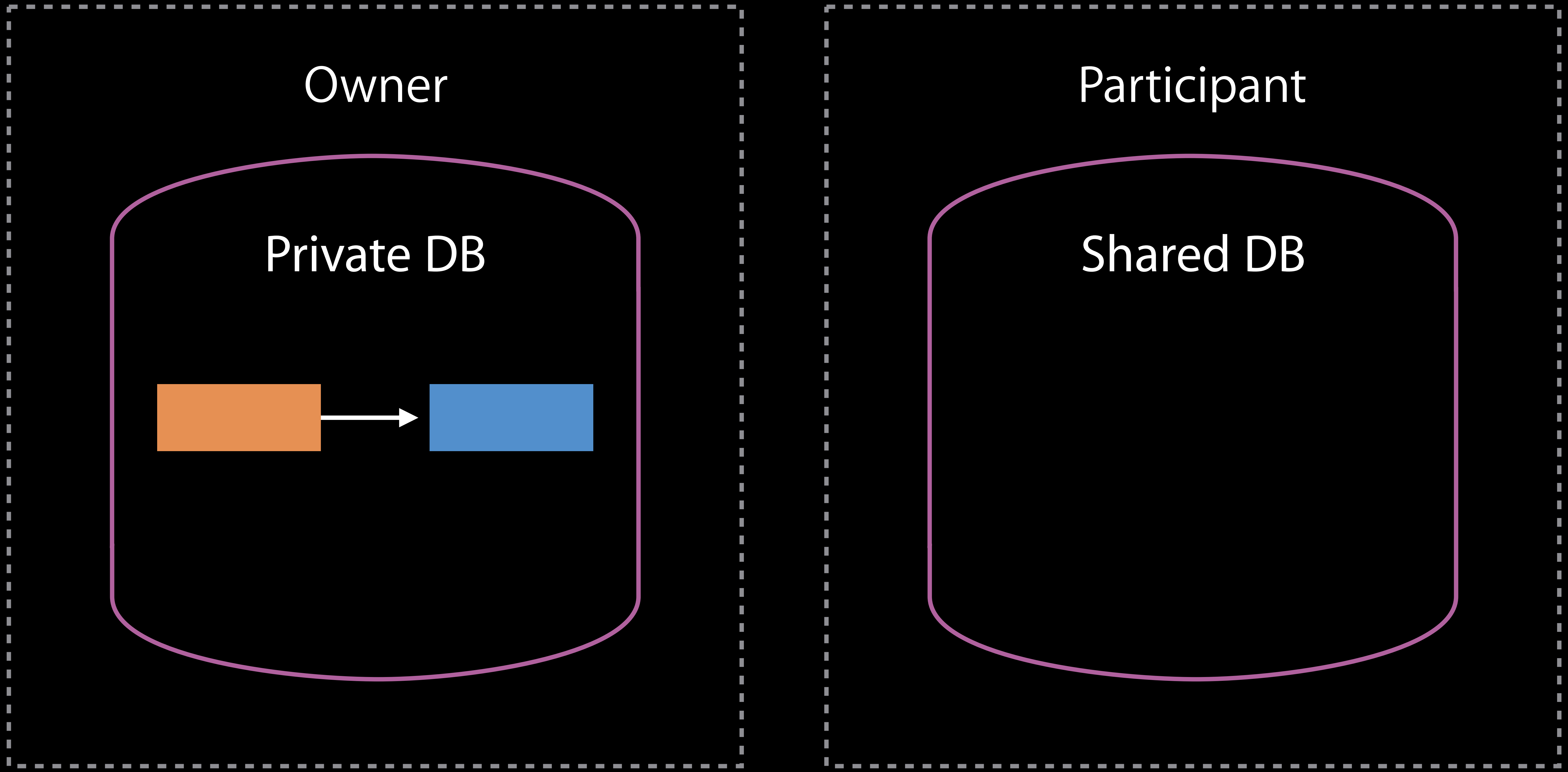


CKShare

Lifecycle—end the share for a participant

Participant leaves the Share by deleting the Share from their Shared DB

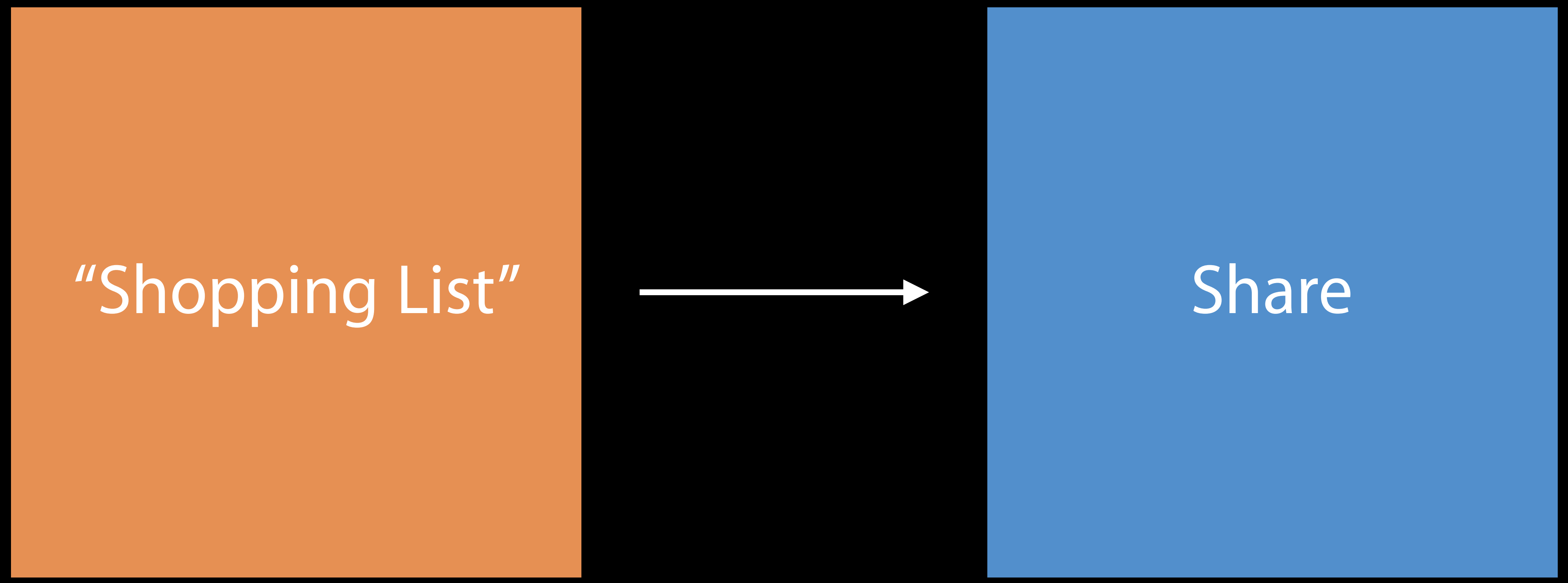
Owner can always remove any Participant



CKShare

Lifecycle—End the share for everyone

Owner deletes the Share from his private DB



CKShare

Lifecycle—End the share for everyone

Owner deletes the Share from his private DB

CKShareParticipant

CKUserIdentity

Participant

```
public class CKShareParticipant {
  public var userIdentity: CKUserIdentity
  …
}

 public class CKUserIdentity {
   public var lookupInfo: CKUserIdentityLookupInfo
   public var nameComponents: PersonNameComponents
 }
```

CKShareParticipant

CKUserIdentity

Participant

```
public class CKShareParticipant {
  public var userIdentity: CKUserIdentity
  …
}

public class CKUserIdentity {
  public var lookupInfo: CKUserIdentityLookupInfo
  public var nameComponents: PersonNameComponents
}
```

CKShareParticipant

CKUserIdentity

Participant

```
public class CKShareParticipant {
   public var userIdentity: CKUserIdentity
   …
}

 public class CKUserIdentity {
   public var lookupInfo: CKUserIdentityLookupInfo
   public var nameComponents: PersonNameComponents
 }
```

CKShareParticipant

CKUserIdentity

Participant

```
public class CKShareParticipant {
  public var userIdentity: CKUserIdentity
  …
}

public class CKUserIdentity {
  public var lookupInfo: CKUserIdentityLookupInfo
  public var nameComponents: PersonNameComponents
}
```

CKShareParticipant

Mapped to iCloud accounts

CKShareParticipant

Mapped to iCloud accounts

Participant #1

```
userIdentity.lookupInfo: <email>
```

Participant #2

```
userIdentity.lookupInfo: <phone>
```

Participant #3

```
userIdentity.lookupInfo: <email>
```

Participant #4

```
userIdentity.lookupInfo: <phone>
```

CKShareParticipant

Mapped to iCloud accounts

Participant #1

```
userIdentity.lookupInfo: <email>
```

Participant #2

```
userIdentity.lookupInfo: <phone>
```

Participant #3

```
userIdentity.lookupInfo: <email>
```

Participant #4

```
userIdentity.lookupInfo: <phone>
```

CKShareParticipant

Mapped to iCloud accounts

Participant #1

```
userIdentity.lookupInfo: <email>
```

Participant #2

```
userIdentity.lookupInfo: <phone>
```

Participant #3

```
userIdentity.lookupInfo: <email>
```

No iCloud account

Verification flow to prove email ownership

Participant #4

```
userIdentity.lookupInfo: <phone>
```

No iCloud account

Verification flow to prove phone ownership

Sharing APIs

If you want to create your own Custom UI

On behalf of the owner

- Setting up the Share

On behalf of the participant

- Accept the Share

watchOS and tvOS

- Shared records available, but no System UI

Owner Sets Up the Share

Adding participants

Owner Sets Up the Share

Adding participants

CKFetchShareParticipantsOperation

- Can look up via
    - Email
    - Phone
    - CloudKit User Record ID

Owner Sets Up the Share

Adding participants

CKFetchShareParticipantsOperation

- Can look up via
  - Email
  - Phone
  - CloudKit User Record ID
- Returns CKShareParticipants

Owner Sets Up the Share

Adding participants

CKFetchShareParticipantsOperation

- Can look up via
  - Email
  - Phone
  - CloudKit User Record ID
- Returns CKShareParticipants

Pass CKShareParticipants to addParticipant

Owner Sets Up the Share

Adding participants

CKFetchShareParticipantsOperation

- Can look up via
  - Email
  - Phone
  - CloudKit User Record ID
- Returns CKShareParticipants

Pass CKShareParticipants to addParticipant

Call CKModifyRecordsOperation to save the share

Participant Accepts a Share

Fetch Share Metadata, then Accept the Share

CKFetchShareMetadataOperation

- Converting a URL to CKShareMetadata

Pass CKShareMetadata to CKAcceptSharesOperation

Participant Accepts a Share

Limitations

No nameComponents, for privacy reasons

```
public class CKUserIdentity {
  public var nameComponents: PersonNameComponents // empty
}
```

Verification flow only available via System UI:

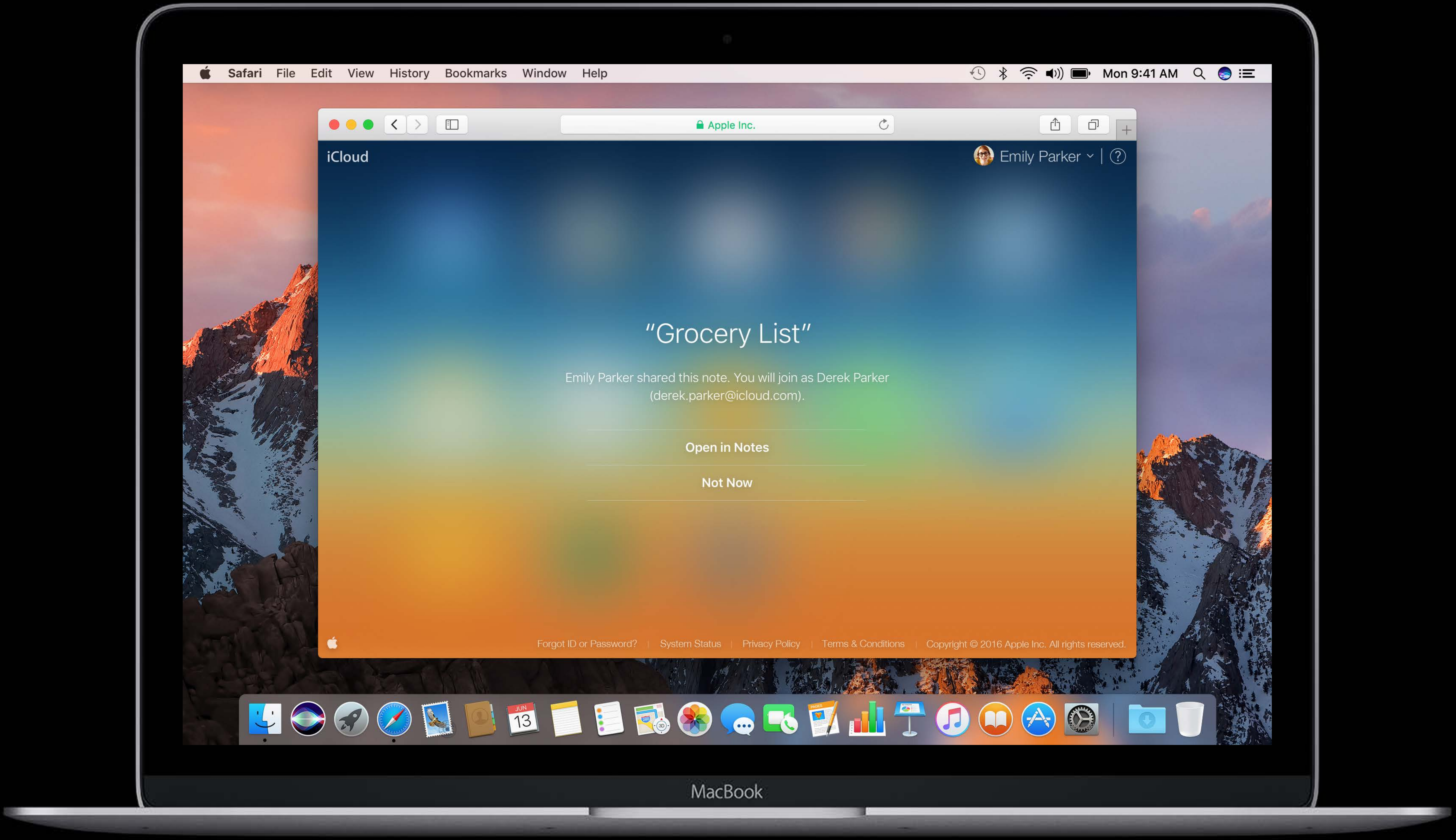
```
CKErrorParticipantMayNeedVerification

shareParticipant.userIdentity.lookupInfo.hasiCloudAccount
```

Sharing

Invitees on older platforms

Owner can invite anybody

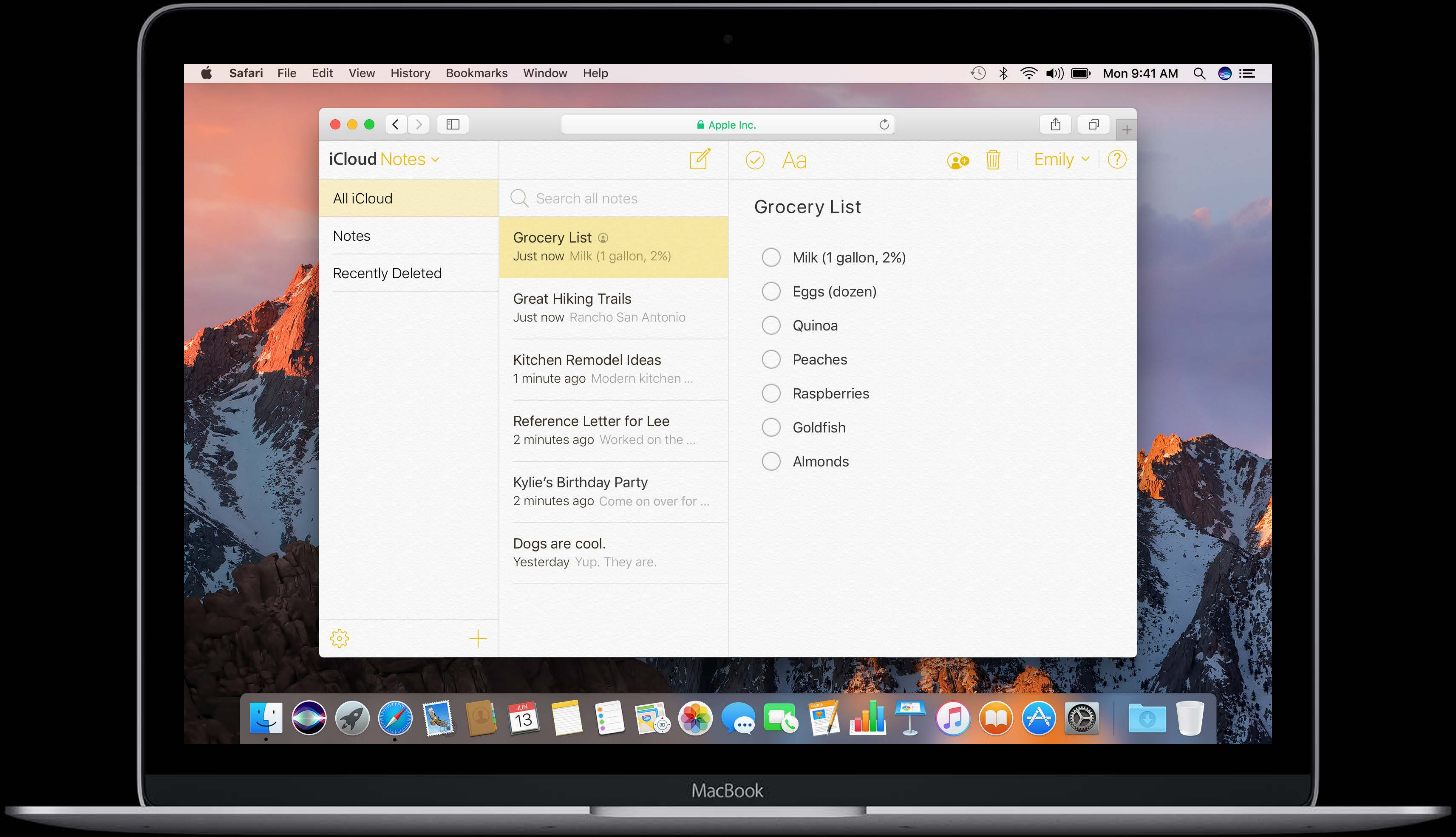
- Invitee may not have installed the latest operating system
- Invitee may not have an Apple product


Safari File Edit View History Bookmarks Window Help
Mon 9:41 AM
Apple Inc.
iCloud
Emily Parker

"Grocery List"

Emily Parker shared this note. You will join as Derek Parker (derek.parker@icloud.com).

Open in Notes
Not Now
Forgot ID or Password? | System Status | Privacy Policy | Terms & Conditions | Copyright © 2016 Apple Inc. All rights reserved.
MacBook



Grocery List
Just now Milk (1 gallon, 2%)

Great Hiking Trails
Just now Rancho San Antonio

Kitchen Remodel Ideas
1 minute ago Modern kitchen ...

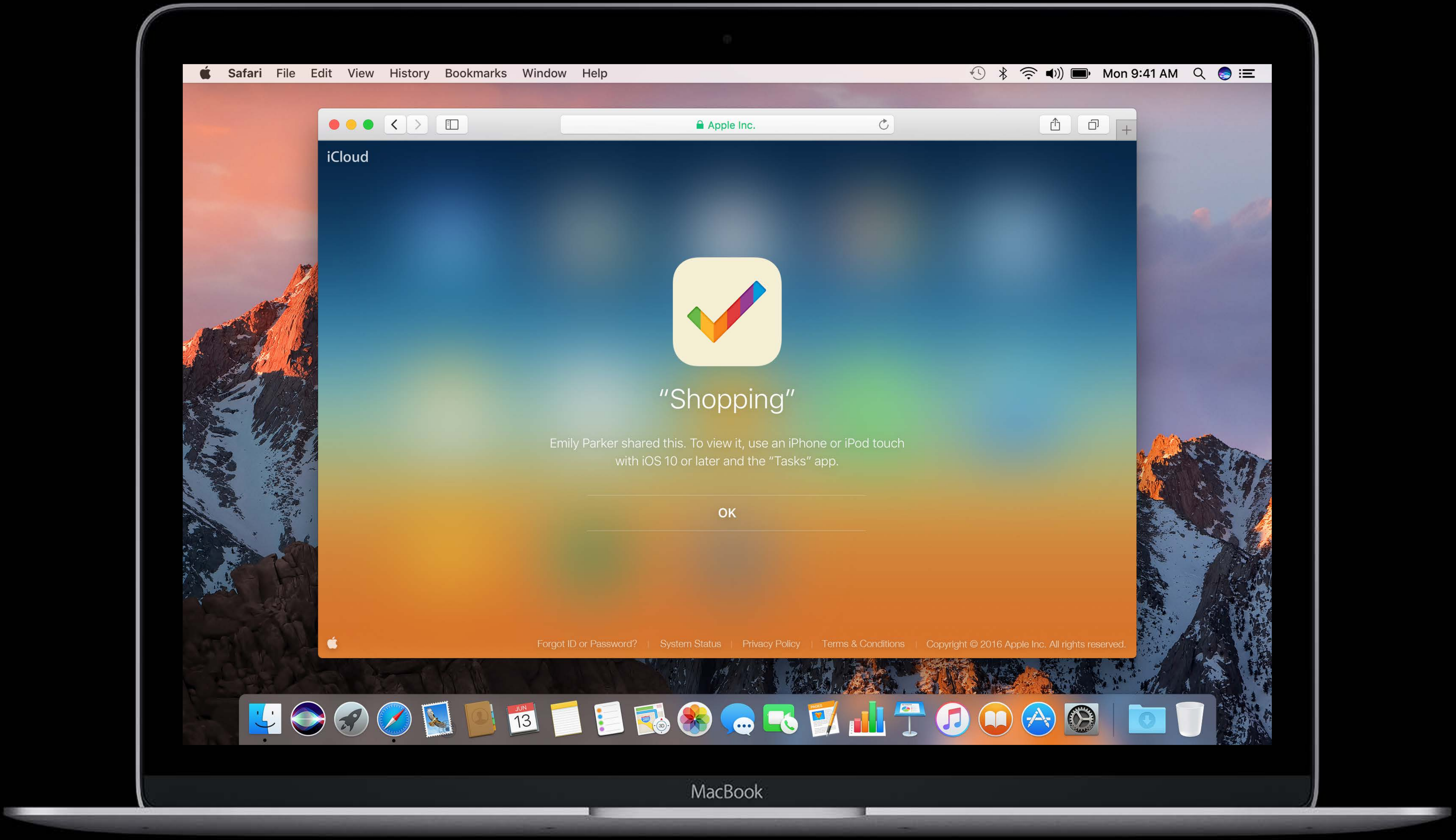
Reference Letter for Lee
2 minutes ago Worked on the ...

Kylie’s Birthday Party
2 minutes ago Come on over for ...

Dogs are cool.
Yesterday Yup. They are.

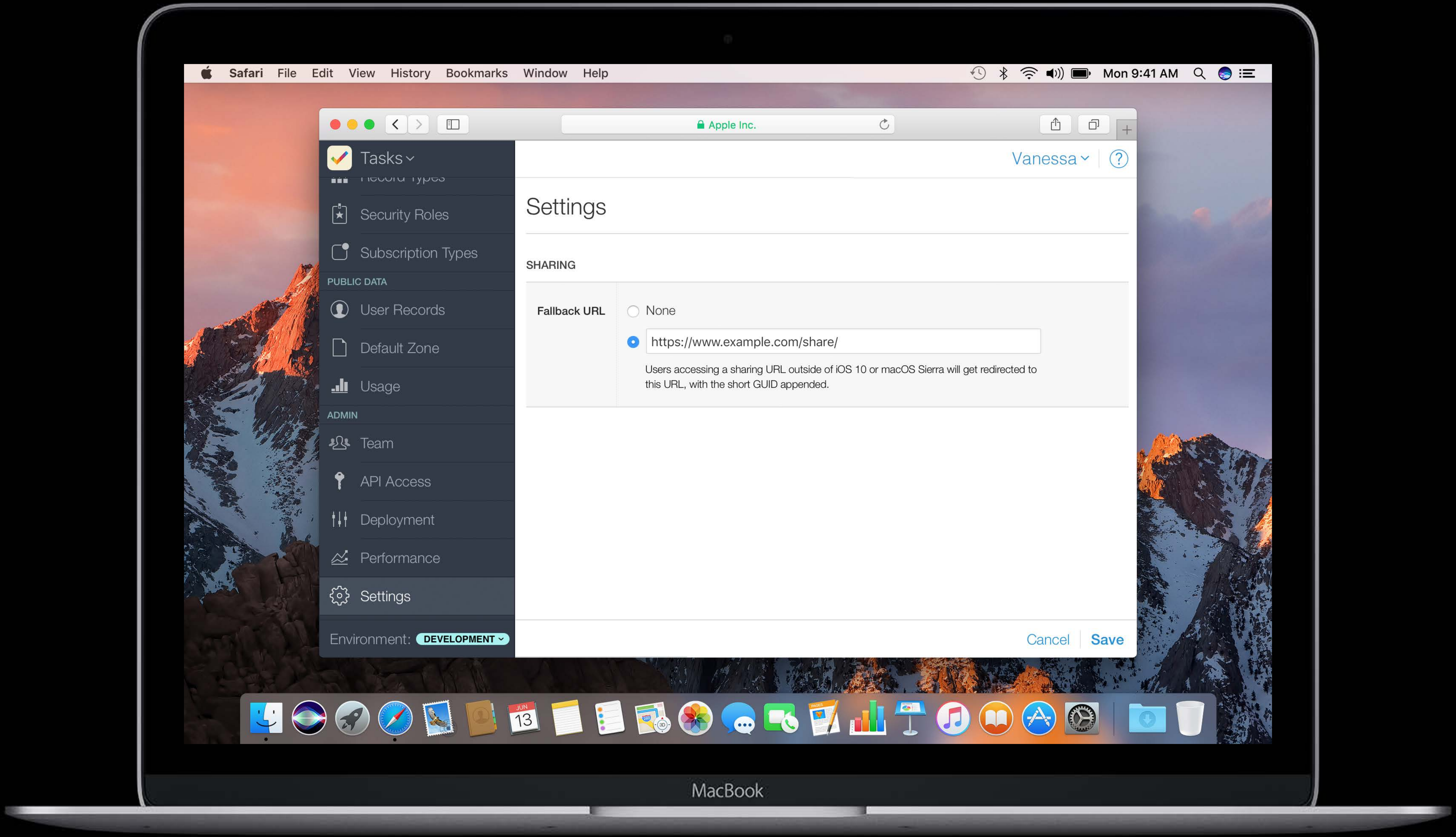
Grocery List

- ○ Milk (1 gallon, 2%)
- ○ Eggs (dozen)
- ○ Quinoa
- ○ Peaches
- ○ Raspberries
- ○ Goldfish
- ○ Almonds


Safari File Edit View History Bookmarks Window Help
Mon 9:41 AM
Apple Inc.
iCloud
OK
Forgot ID or Password? | System Status | Privacy Policy | Terms & Conditions | Copyright © 2016 Apple Inc. All rights reserved.
MacBook

"Shopping"

Emily Parker shared this. To view it, use an iPhone or iPod touch with iOS 10 or later and the "Tasks" app.

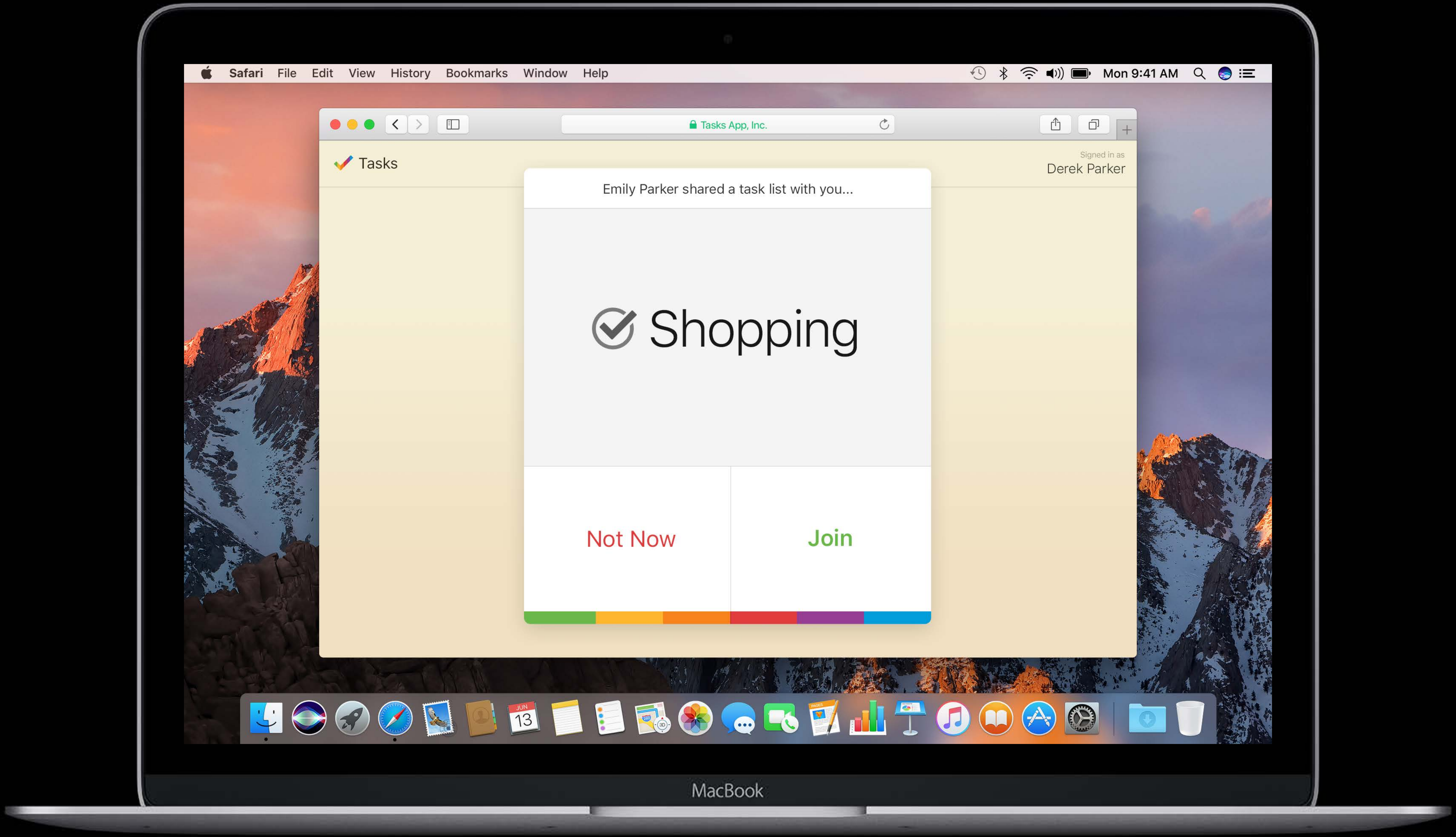


Settings

SHARING

Fallback URL: None [✓ https://www.example.com/share/]

Users accessing a sharing URL outside of iOS 10 or macOS Sierra will get redirected to this URL, with the short GUID appended.


Safari File Edit View History Bookmarks Window Help
Mon 9:41 AM
Tasks App, Inc.
Tasks
Signed in as
Derek Parker
Emily Parker shared a task list with you...
Shopping
Not Now
Join
MacBook

Sharing

Get your container ready for Sharing

CKRecordTypeShare

- Behaves like any other Record Type in CloudKit
- Can create custom fields
- Can run queries

To trigger its creation

- Share a record in a custom zone in any private database in the development environment
- Deploy schema to production

Now, users in production can create Shares

Summary

What’s new is now old

CloudKit is available on all platforms

Telemetry available on CloudKit Dashboard

API Improvements

New Feature—Sharing

- Sharing System UI
- Sharing APIs, Objects, and Lifecycle
- Configure your fallback URL!

More Information

https://developer.apple.com/wwdc16/226

Related Sessions

CloudKit Best Practices	Pacific Heights	Friday 9:00AM

Labs

CloudKit and iCloud Lab	Frameworks Lab D	Thursday 4:00PM
CloudKit and iCloud Lab	Frameworks Lab B	Friday 12:00PM

WWDC16