SAP Cloud Application Programming Model

https://cap.cloud.sap

Roundtable #3

October 5th, 2022

PUBLIC

Today's hosts:

Sebastian Schmidt

Michael Hellenschmidt

THE BEST RUN SAP

Agenda

Introduction: Recap from last roundtable	05 min
Demo #1	20 min
▪ New Extensibility Guide – Christian Georgi	
Presentation #2	20 min
▪ Upsert Support by CDS Runtimes – Adrian Görler	
Open discussion	45 min

Please introduce yourself and interact with your peers

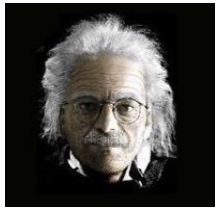
Please give a brief introduction

- In the chat:
  - Name
  - Company
  - Location
  - Role
  - What are you looking for from this group?
  - Your experience with CAP

We are providing access to the people that drive product development

CAP Leads

Ole Lilienthal
Head of unit



Daniel Hutzel
CPO

Product Owners		Areas
	Steffen Weinstock	CDS / Compiler
	Johannes Vogel	Node.js runtime
	Matthias Braun	Java runtime
	Adrian Görler	
	Christian Georgi	Tools

Summary of Second CAP Customer Roundtable in July

We had:

- reCAP hybrid 2022 - Summary (Sebastian Schmidt)
- Demo #1 Deployment to Kyma runtime (Uwe Klinger)
- Demo #2 Details on Streamlined MT and some outlook on extensibility feature (Daniel Hutzel / Eckart Liemke
- Q&A

New Extensibility Guide

Following the new multitenancy guide,

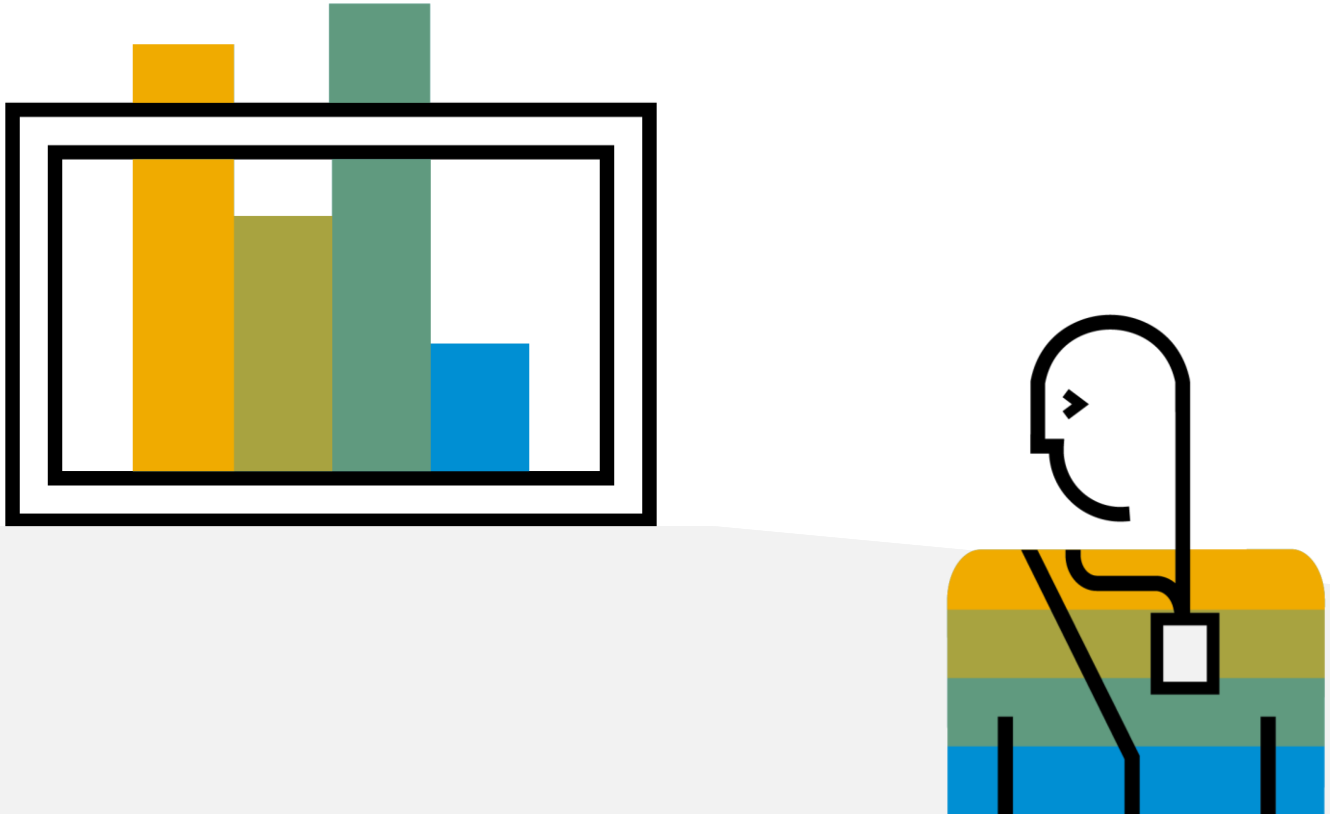
there is now a new guide for extending SaaS apps.

It features

- Easier setup of extension projects
- Local development roundtrips with SQLite
- More examples for extending CDS models

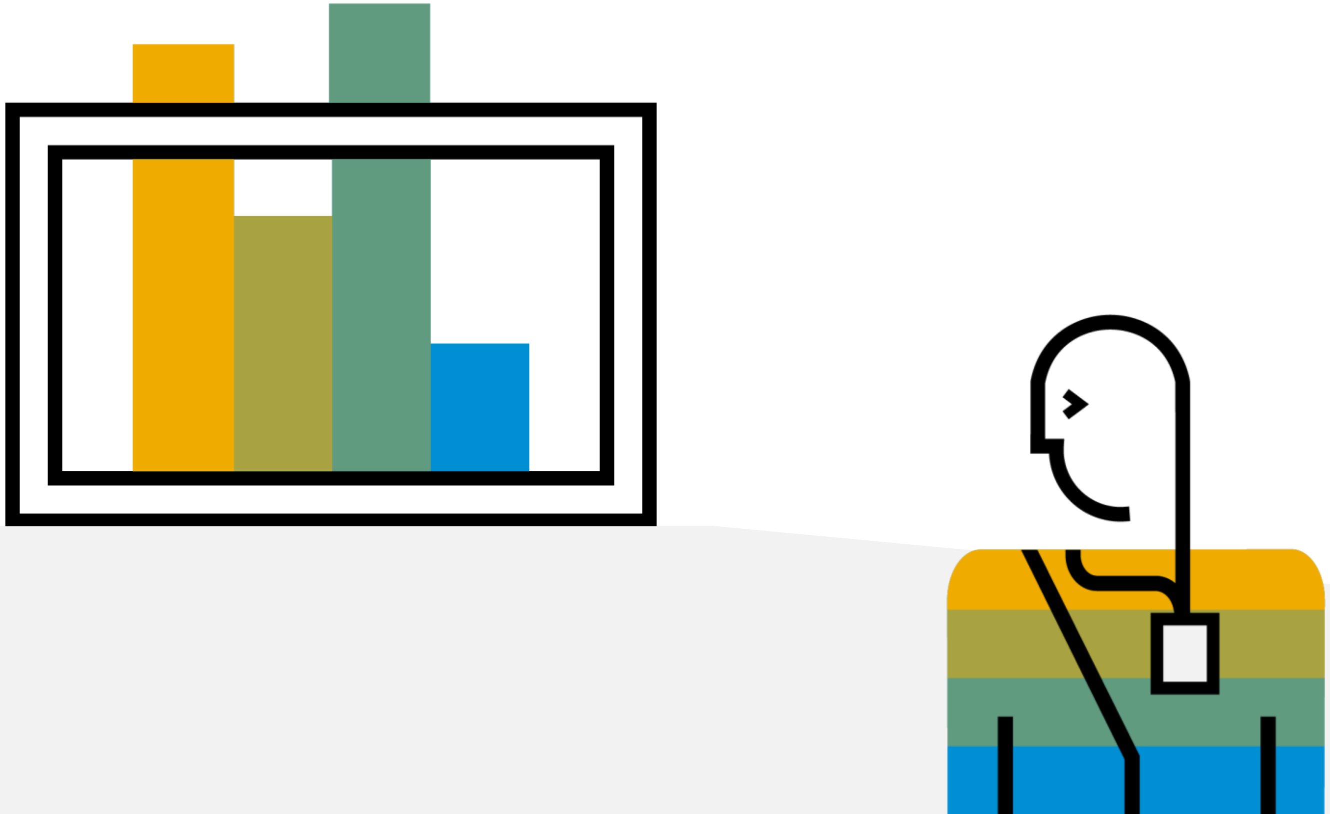
Demo #1

- New Extensibility Guide – Christian Georgi

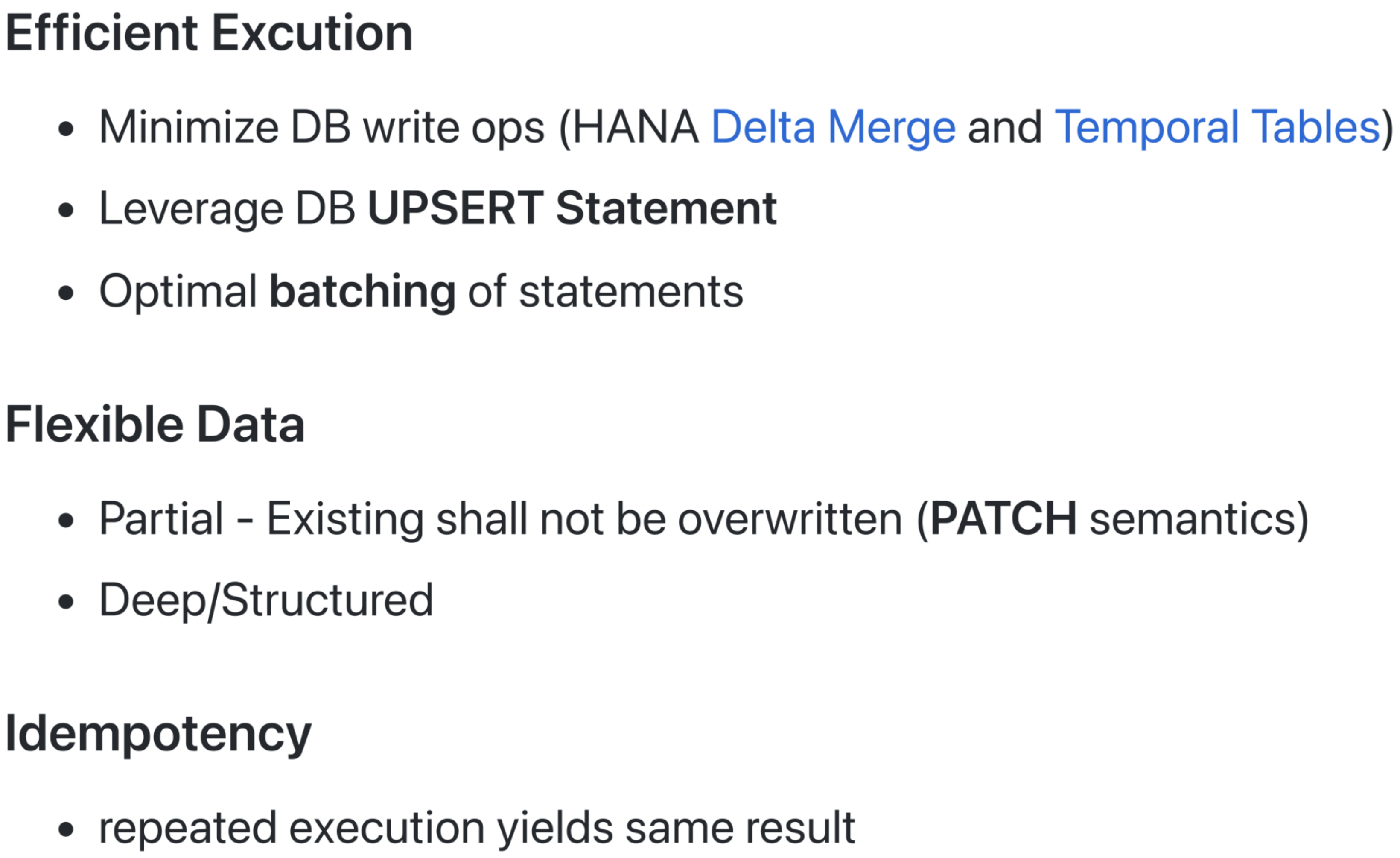


Demo #2

Upsert Support by CDS Runtimes – Adrian Görler



Requirements



Efficient Excution

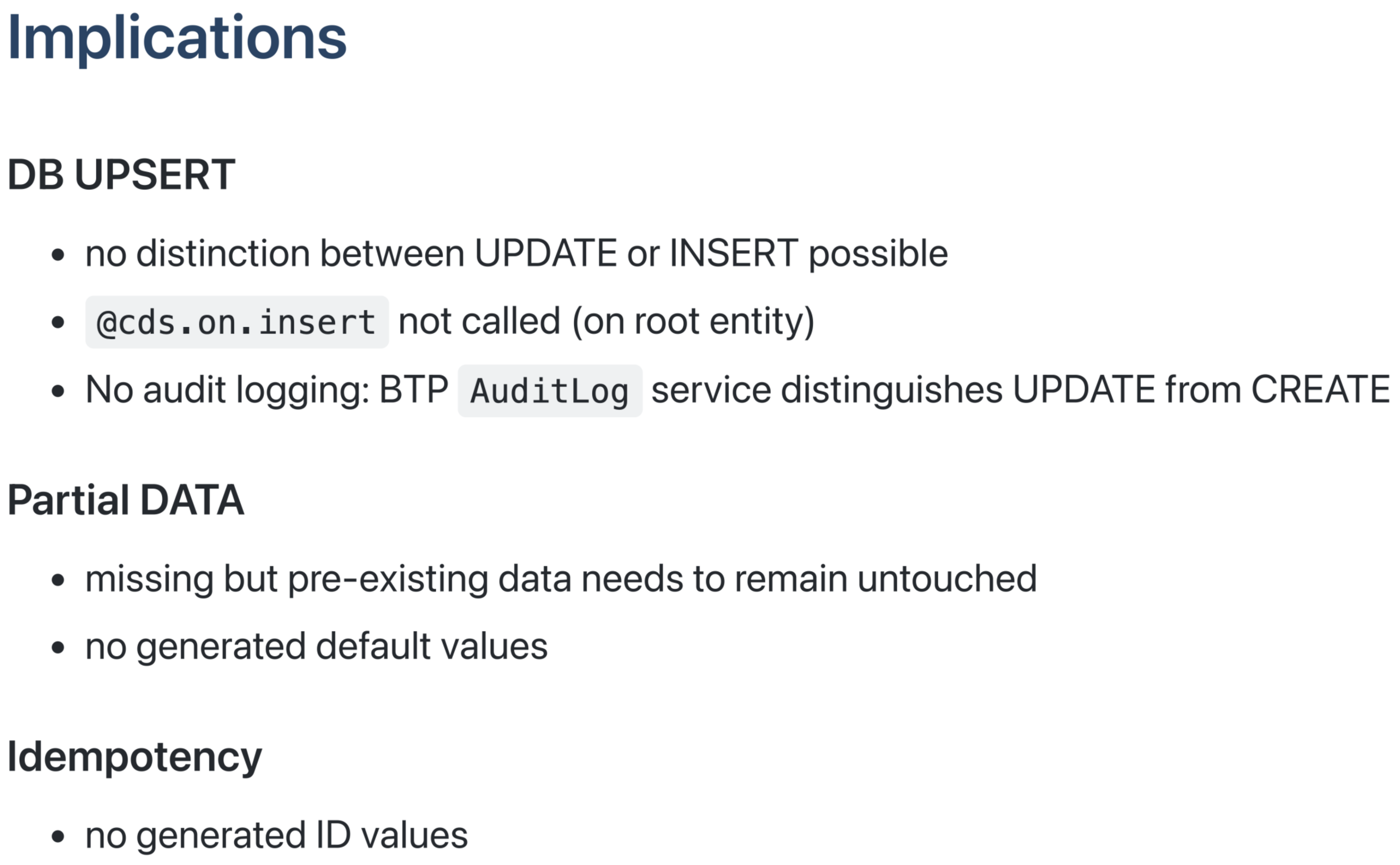
- Minimize DB write ops (HANA Delta Merge and Temporal Tables)
- Leverage DB **UPSERT Statement**
- Optimal **batching** of statements

Flexible Data

- Partial - Existing shall not be overwritten (**PATCH** semantics)
- Deep/Structured

Idempotency

- repeated execution yields same result



Implications

DB UPSERT

- no distinction between UPDATE or INSERT possible
- `@cds.on.insert` not called (on root entity)
- No audit logging: BTP `AuditLog` service distinguishes UPDATE from CREATE

Partial DATA

- missing but pre-existing data needs to remain untouched
- no generated default values

Idempotency

- no generated ID values

Basic Concept

DB UPSERT made availabe in CDS QL

`UPSERT` Builder

```
Upsert.into(<ref>).entries(<data array>) // Java
```

```
UPSERT.into(<ref>).entries(<data array>) // Node.js
```

ID values are provided in the data!

Similar to

```
Update.entity(<ref>).entries(<data array>) // Java; ID values in data
Insert.into(<ref>).entries(<data array>)   // Java
```

```
INSERT.into(<ref>).entries(<data array>)   // Node.js
```

`UPSERT` w/ single entry

Similar to `UPDATE` but entity is created if it doesn't exist in db

```
UPSERT.into('Books').entries(
  { ID: 42, title: 'CAP', year: 2022 }
)
```

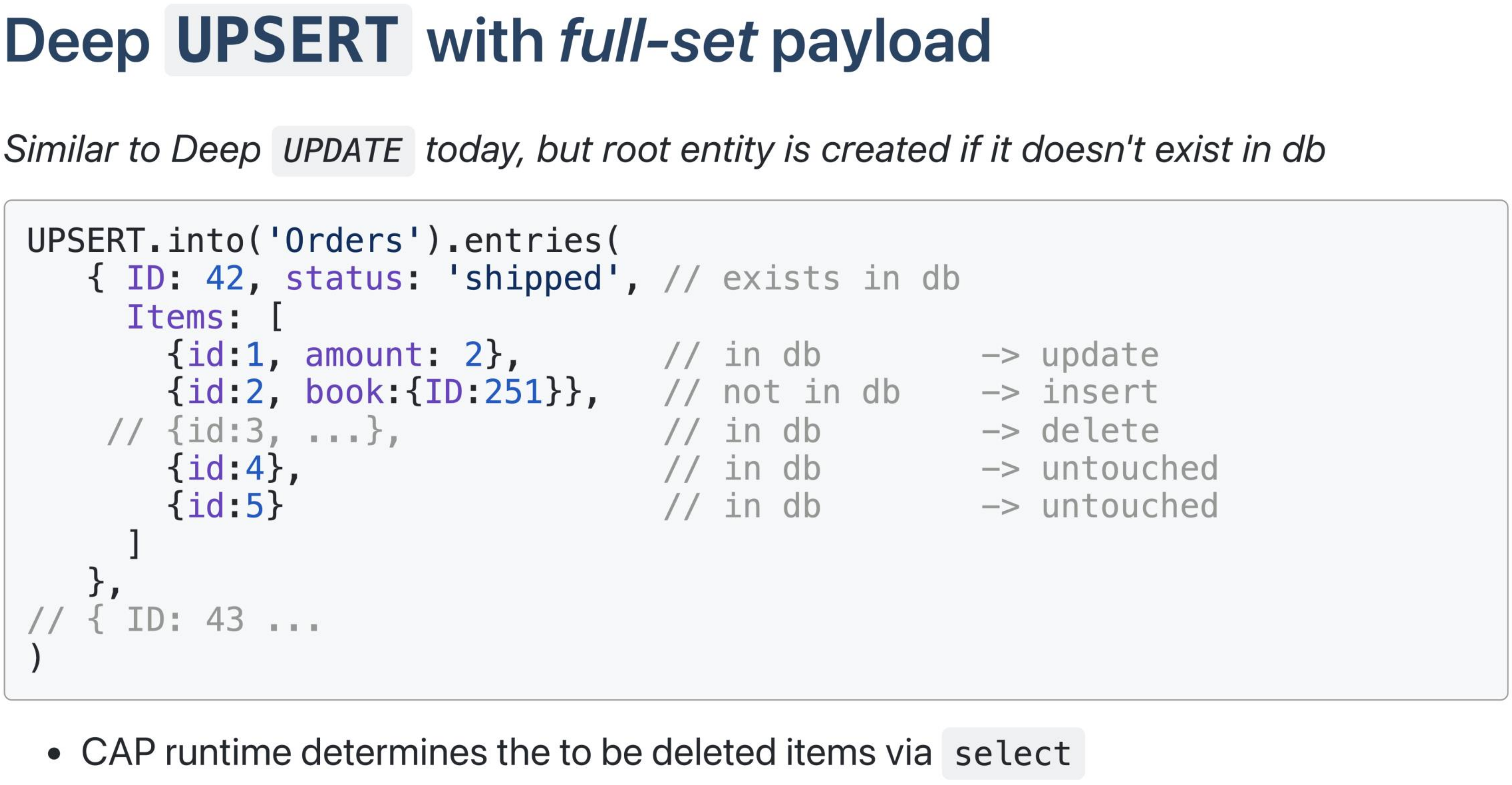
```
UPSERT BOOKS (ID, title, year) VALUES (42, 'CAP', 2022) WITH PRIMARY KEY;  -- HANA
```

UPSERT w/ many entries (bulk)

```
UPSERT.into('Books').entries(
  { ID: 42, title: 'CAP', stock: 3 },
  { ID: 43, title: 'CDS 5', year: 2021 },
  { ID: 44, title: 'CDS 6', year: 2022 }
)
```

Efficient **batch execution** for entries w/ same set of elements (43, 44):

```
UPSERT BOOKS (ID, title, stock) VALUES (?, ?, ?) WITH PRIMARY KEY;  -- 42

UPSERT BOOKS (ID, title, year)  VALUES (?, ?, ?) WITH PRIMARY KEY;  -- 43,44
```



Deep `UPSERT` with *full-set* payload

Similar to Deep `UPDATE` today, but root entity is created if it doesn't exist in db

```
UPSERT.into('Orders').entries(
   { ID: 42, status: 'shipped', // exists in db
     Items: [
       {id:1, amount: 2},       // in db        -> update
       {id:2, book:{ID:251}},   // not in db    -> insert
    // {id:3, ...},             // in db        -> delete
       {id:4},                  // in db        -> untouched
       {id:5}                   // in db        -> untouched
     ]
   },
// { ID: 43 ...
)
```

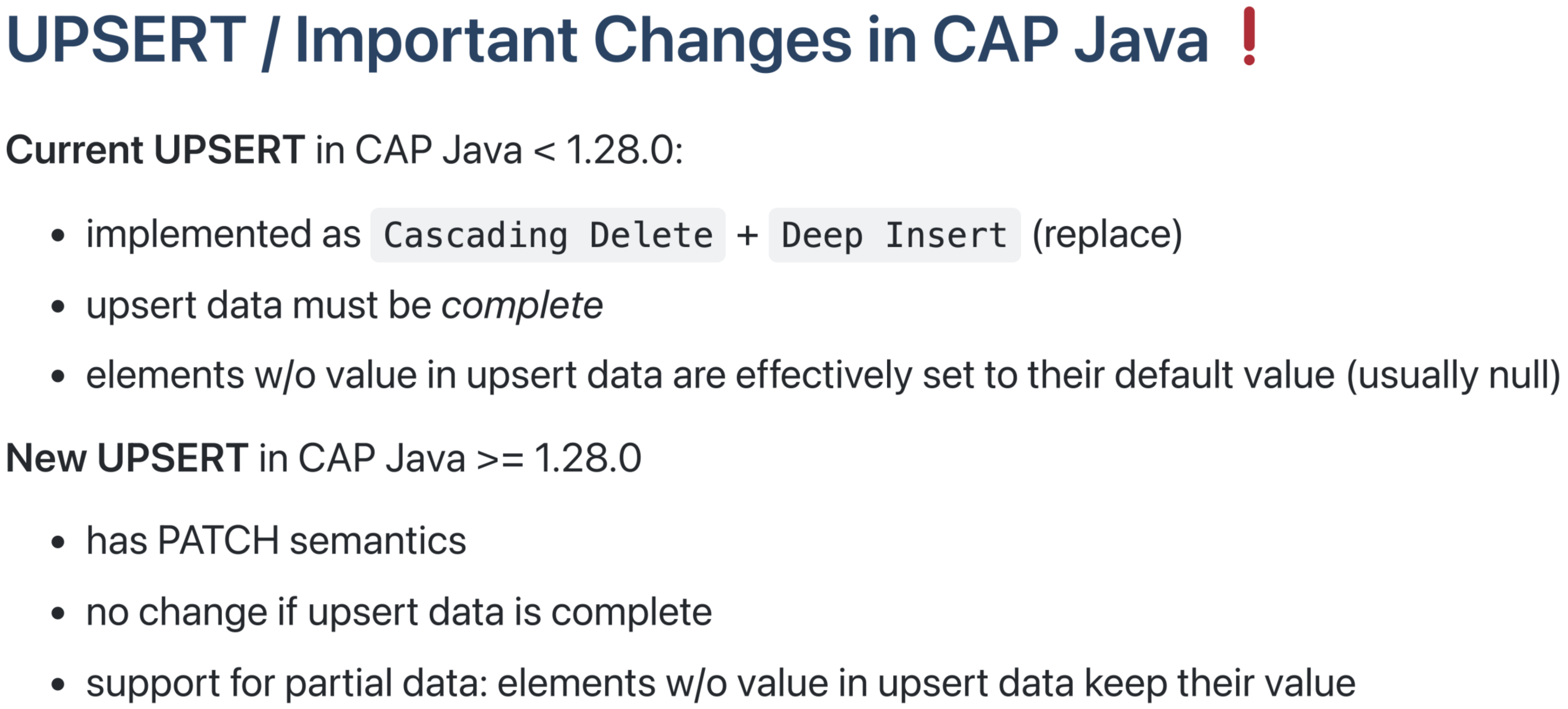
- CAP runtime determines the to be deleted items via `select`

Use Shallow `UPSERT` for Insert-only

```
UPSERT.into('Employees').entries([
  { ID: 1, name: 'Bob',   skills : [
    { ID : 100, skill : 'paint'},
    { ID : 101, skill : 'drill'}]},
  { ID: 2, name: 'Alice', skills : [{ ID: 200, skill : 'think' }] }])
```

Consider *splitting* into two shallow uperts:

```
UPSERT.into('Employees').entries([
  { ID: 1, name: 'Bob' },
  { ID: 2, name: 'Alice' }])

UPSERT.into('Skills').entries([
  { ID : 100, employee : { ID : 1}, skill : 'paint' },
  { ID : 101, employee : { ID : 1}, skill : 'drill' },
  { ID : 200, employee : { ID : 2}, skill : 'think' }])
```



UPSERT / Important Changes in CAP Java ❗

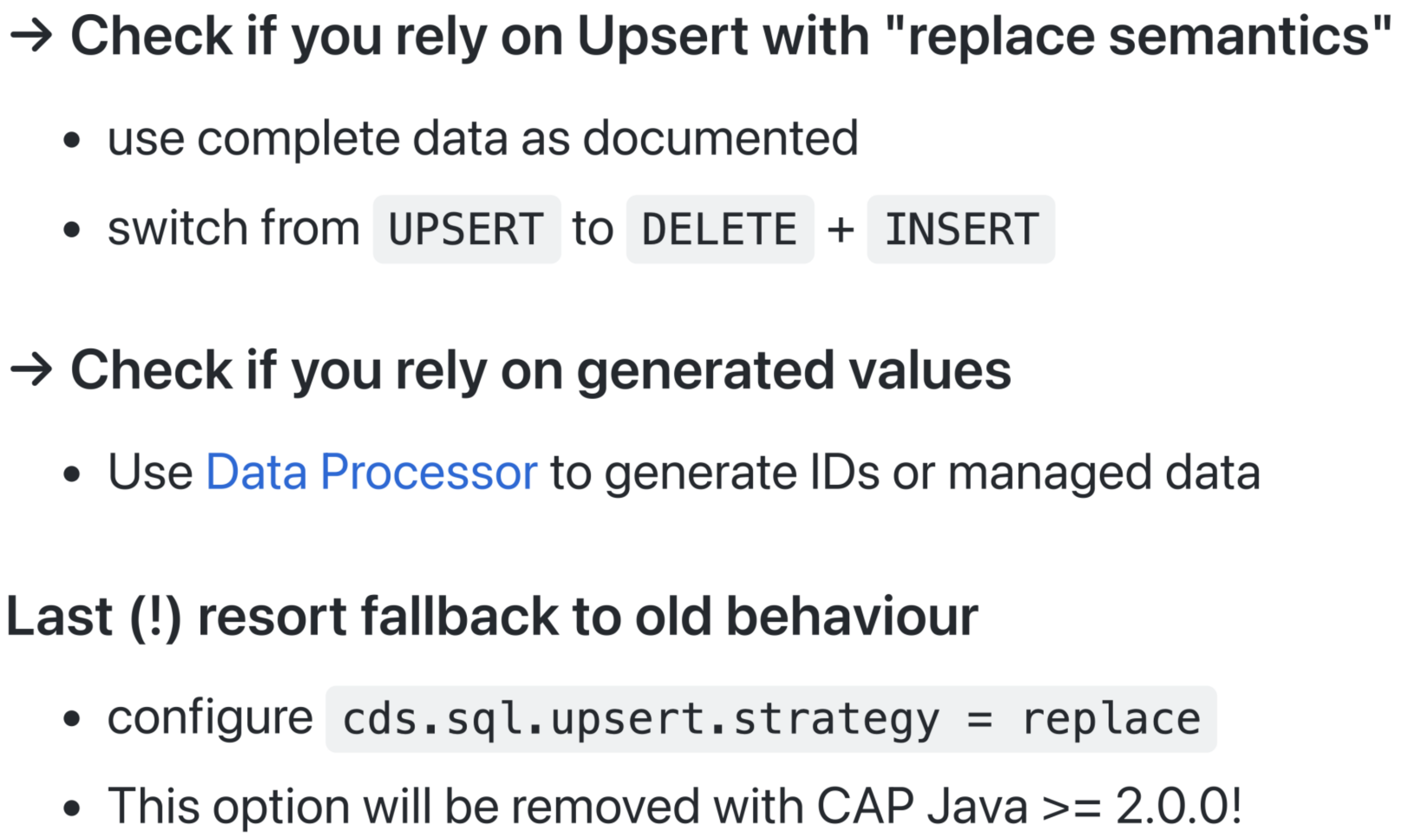
Current UPSERT in CAP Java < 1.28.0:

- implemented as `Cascading Delete` + `Deep Insert` (replace)
- upsert data must be *complete*
- elements w/o value in upsert data are effectively set to their default value (usually null)

New UPSERT in CAP Java >= 1.28.0

- has PATCH semantics
- no change if upsert data is complete
- support for partial data: elements w/o value in upsert data keep their value

Adapt Java Code to New Behavior



→ Check if you rely on Upsert with "replace semantics"

- use complete data as documented
- switch from `UPSERT` to `DELETE` + `INSERT`

→ Check if you rely on generated values

- Use Data Processor to generate IDs or managed data

Last (!) resort fallback to old behaviour

- configure `cds.sql.upsert.strategy = replace`
- This option will be removed with CAP Java >= 2.0.0!

Outlook: Delta Payloads

OData - Related Entities as Delta Payload

Use a nested delta representation to update related entities

```
PUT/PATCH /CatalogService/Orders

[
  {
    "@type": "#CatalogService.Orders",
    "ID": 42,
    "status" : "shipped",
    "items@delta": [
      { "id": 1, "amount" : 2 },
      { "id": 2, "book" : { ID : 251 } },
      { "@removed": { "reason": "deleted" }, "id": 3 }
    ]
  },
  // { "ID" : 43 ...
]
```

Deep UPSERT with *Delta* Payload

Delta payloads as in OData 4.01 tell the CAP runtime what to do for efficient execution:

```
UPSERT.into('Orders').entries(
   { ID: 42, status: 'shipped',
     Items: { @delta:[       // delta payload
       { id:1, amount: 2 },     // upsert (default)
       { id:2, book:{ID:251} }, // upsert (default)
       { id:3, @removed:true }  // delete
     ]}
   },
// { ID: 43 ...
)
```

- Support in Deep `UPSERT` & `UPDATE`
- CAP runtime accepts *any* value for `@removed`
- Possible optimization: `@PreviousState` : `existed` -> UPDATE

OData - Update an Entity Collection with Delta Payloads

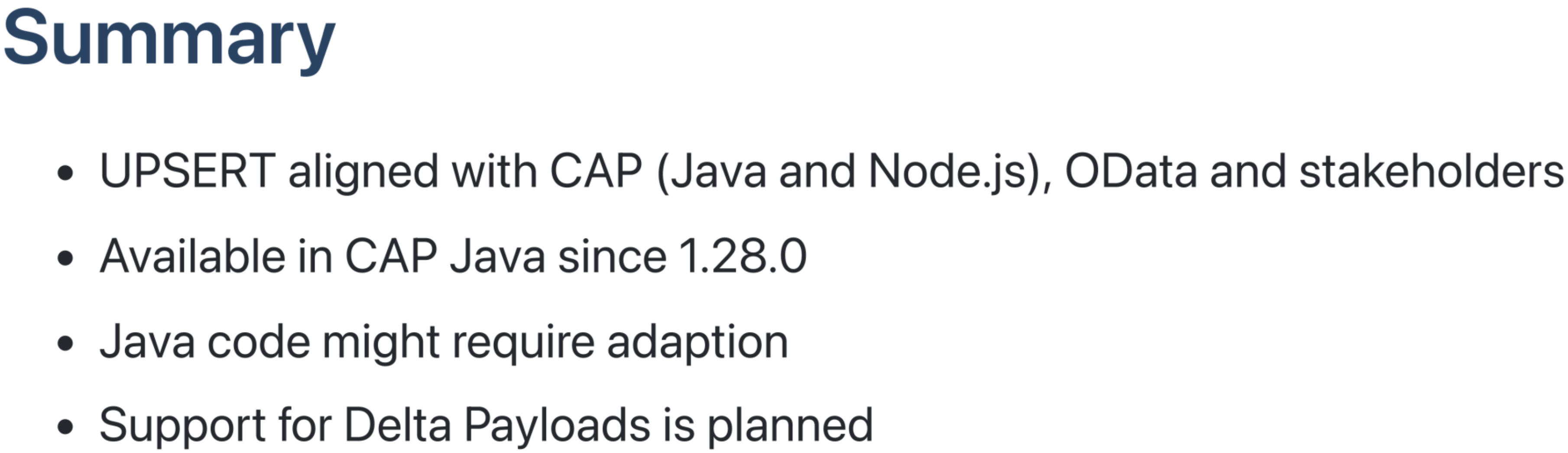
Use a delta payload to update a collection of entities

```
PATCH /CatalogService/Books

[
  { "ID" : 42, "title": "CAP", "stock": 3 },
  { "ID" : 43, "title": "CDS 5", "year": 2021 },
  { "ID" : 45, "@removed" : { "reason" : "deleted } }
]
```

will be translated into a bulk UPSERT and a DELETE

```
UPSERT.into('Books').entries(
  { ID: 42, title: 'CAP', stock: 3 },
  { ID: 43, title: 'CDS 5', year: 2021 })

DELETE.from('Books').where('ID = ', 45)
```

Much more efficient then $batch request 😃

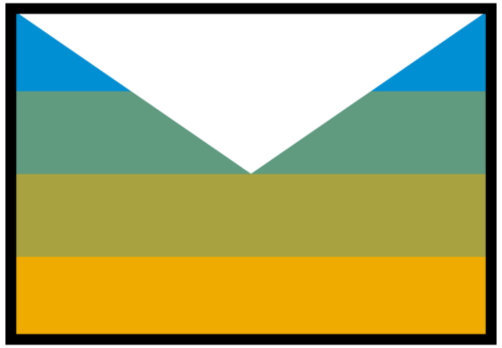


Summary

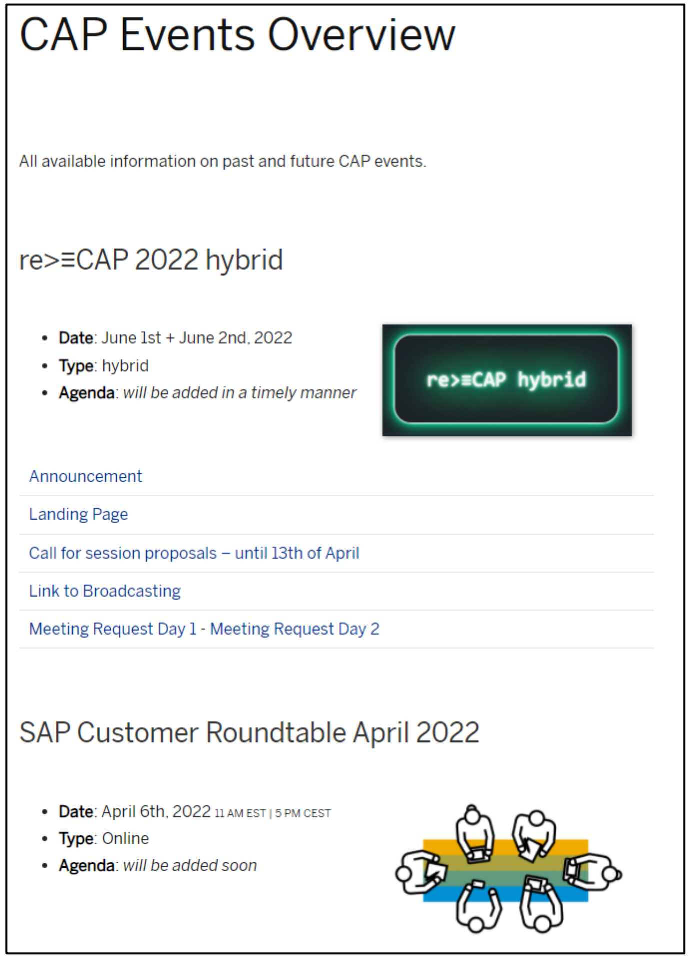
- UPSERT aligned with CAP (Java and Node.js), OData and stakeholders
- Available in CAP Java since 1.28.0
- Java code might require adaption
- Support for Delta Payloads is planned

Stay in touch!

Invitations to CAP events via email?



michael.hellenschmidt@sap.com
se.schmidt@sap.com



CAP Events Overview

All available information on past and future CAP events.

re>≡CAP 2022 hybrid

- **Date**: June 1st + June 2nd, 2022
- **Type**: hybrid
- **Agenda**: *will be added in a timely manner*

Announcement

Landing Page

Call for session proposals – until 13th of April

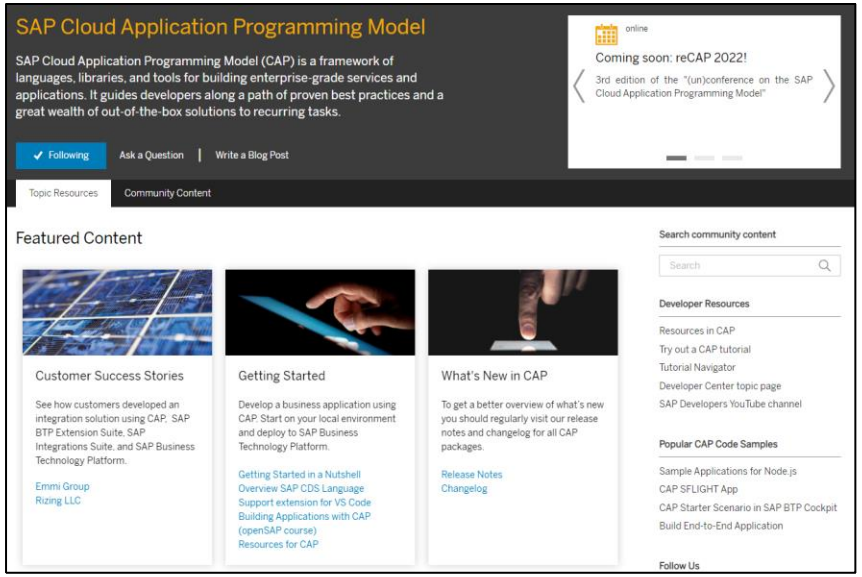
Link to Broadcasting

Meeting Request Day 1 - Meeting Request Day 2

SAP Customer Roundtable April 2022

- **Date**: April 6th, 2022 11 AM EST | 5 PM CEST
- **Type**: Online
- **Agenda**: *will be added soon*

https://cap.cloud.sap/docs/events/



SAP Cloud Application Programming Model

SAP Cloud Application Programming Model (CAP) is a framework of languages, libraries, and tools for building enterprise-grade services and applications. It guides developers along a path of proven best practices and a great wealth of out-of-the-box solutions to recurring tasks.

online

Coming soon: reCAP 2022!

3rd edition of the "(un)conference on the SAP Cloud Application Programming Model"

Featured Content

Customer Success Stories

See how customers developed an integration solution using CAP, SAP BTP Extension Suite, SAP Integrations Suite, and SAP Business Technology Platform.

Emmi Group
Rizing LLC

Getting Started

Develop a business application using CAP. Start on your local environment and deploy to SAP Business Technology Platform.

Getting Started in a Nutshell
Overview SAP CDS Language
Support extension for VS Code
Building Applications with CAP (openSAP course)
Resources for CAP

What's New in CAP

To get a better overview of what's new you should regularly visit our release notes and changelog for all CAP packages.

Release Notes
Changelog

Search community content

Developer Resources

Resources in CAP
Try out a CAP tutorial
Tutorial Navigator
Developer Center topic page
SAP Developers YouTube channel

Popular CAP Code Samples

Sample Applications for Node.js
CAP SFLIGHT App
CAP Starter Scenario in SAP BTP Cockpit
Build End-to-End Application

Follow Us

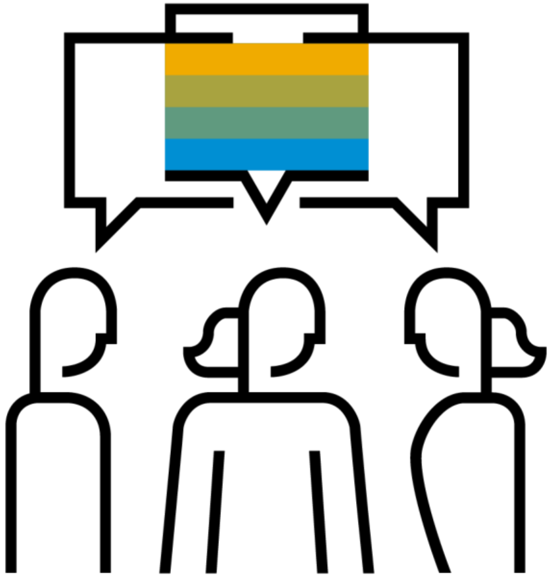
https://community.sap.com/topics/cloud-application-programming

Q&A

Open discussion – what's on your mind regarding CAP?

Possible topics:

- What are you working on?
- What are you interested in learning more about?
- What feedback do you have on our products?
  - What do you like?
  - What do you need CAP to do that it does not?
  - What new capabilities do you need from CAP?
- Are you stuck in your projects?
- Would you like to demo what they you working on?



Thank you.

Contact information:

Sebastian Schmidt
Development Manager
se.schmidt@sap.com

Michael Hellenschmidt
Delivery Head
michael.hellenschmidt@sap.com