📗 CDS Annotation Playbook — 20 Real-Time "Recipes" for ABAPers

Think of these as **ready-to-deploy design templates** — each combines 3–6 annotations that, together, **eliminate large ABAP workarounds** in real-life projects.
They're grouped by **use case category**, so you can pick and apply them instantly.

🧱 A. Security & Authorization Recipes

🔒 1. Role-Based Authorization + OData Exposure

```
@AccessControl.authorizationCheck: #CHECK
@OData.publish: true
@EndUserText.label: 'Sales Order Data'
define view entity ZI_SalesAuth as select from vbak { vbeln, vkorg, netwr }
```

✅ **Replaces:**

- `AUTHORITY-CHECK` logic
- SEGW model & MPC_EXT coding
  💼 **Use Case:** Secure Fiori List Report for sales headers.

🔐 2. Data Masking + Restricted Access

```
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Employee Sensitive Data'
@UI.hidden: true
define view entity ZI_Employee as select from pa0001 { pernr, name, salary
}
```

✅ **Replaces:**

- Masking logic in ALV/Fiori
  💼 **Use Case:** HR data exposure — hide salary for unauthorized users.

🧍 3. Audit + User Tracking

```
@Semantics.systemDate.createdAt: true
@Semantics.user.createdBy: true
@Semantics.businessDate.lastChangedAt: true
@Semantics.user.lastChangedBy: true
```

```
define view entity ZI_AuditTrail as select from zorder { vbeln, created_at,
created_by, changed_at, changed_by }
```

✅ **Replaces:**

- `SY-UNAME`, `SY-DATUM` update logic
  💼 **Use Case:** Change audit in Z-tables.

🧮 B. Analytical / Reporting Recipes

📊 4. Sales Summary Cube

```
@ObjectModel.dataCategory: #CUBE
@Analytics.query: true
@DefaultAggregation: #SUM
@EndUserText.label: 'Sales Summary'
define view entity ZI_SalesSummary as
  select from vbak
  group by vkorg
  { vkorg, sum( netwr ) as total_sales }
```

✅ **Replaces:**

- BEx query or report program aggregation
  💼 **Use Case:** Fiori analytical tile – “Sales by Org.”

📈 5. Analytical Query with KPI Visualization

```
@Analytics.query: true
@UI.dataPoint: { title: 'Total Sales', value: 'total_sales' }
@UI.chart: [{ type: #BAR, title: 'Sales by Customer', dimensions:
['kunnr'], measures: ['total_sales'] }]
define view entity ZI_SalesChart as
  select from vbak
  group by kunnr
  { kunnr, sum( netwr ) as total_sales }
```

✅ **Replaces:**

- KPI modeling, custom chart UI
  💼 **Use Case:** Real-time Fiori KPI visualization.

📆 6. Dynamic Date Logic (Fiscal Reports)

```
@AnalyticsDetails.query.display: #ON
@DefaultAggregation: #SUM
define view entity ZI_FiscalReport as
  select from vbak
  { vkorg, sum( netwr ) as total_sales, add_days( audat, 30 ) as
next_month_due }
```

✅ **Replaces:**

- ABAP `ADD_DAYS` / date calculation in loops
  💼 **Use Case:** Month-wise performance dashboards.

🧰 C. Performance & Optimization Recipes

⚙️ 7. Buffering + Compare Filter

```
@AbapCatalog.buffering: true
@AbapCatalog.compiler.compareFilter: true
@Environment.systemField: #CLIENT
@ClientDependent: true
define view entity ZI_BufferedData as select from mara { matnr, ersda,
mtart }
```

✅ **Replaces:**

- Manual SELECT caching or buffering logic
  💼 **Use Case:** Heavy read tables like MARA/KNA1.

⚡ 8. Pushdown Ready View + Table Function

```
@AbapCatalog.sqlViewName: 'ZV_TOPCUST'
@AccessControl.authorizationCheck: #NOT_REQUIRED
define table function ZTF_TopCustomers
returns { kunnr: abap.kunnr, total_sales: abap.curr(15,2) }
implemented by method zcl_top_customers=>get_data;
```

✅ **Replaces:**

- AMDP call or nested internal table aggregation
  💼 **Use Case:** Fetch top 10 customers efficiently.

👩 D. UI & Fiori Recipes

□ 9. List Report with Selection Fields

```
@OData.publish: true
@UI.lineItem: [{ position: 10, label: 'Order No' }, { position: 20, label:
'Net Value' }]
@UI.selectionField: [{ position: 10, label: 'Sales Org' }]
@EndUserText.label: 'Sales List Report'
define view entity ZC_SalesList as select from vbak { vbeln, vkorg, netwr }
```

✅ **Replaces:**

- Custom ALV and selection-screen coding
  💼 **Use Case:** Quick List Report app without SEGW or dynpro.

🧭 10. Object Page with Header and Facets

```
@UI.headerInfo: { typeName: 'Sales Order', title: { type: #STANDARD, value:
'vbeln' } }
@UI.facet: [{ id: 'Header', label: 'General Info', type:
#IDENTIFICATION_REFERENCE }]
@UI.lineItem: [{ position: 10, label: 'Customer' }, { position: 20, label:
'Value' }]
@UI.identification: [{ position: 10, label: 'Created On' }]
define view entity ZC_SalesDetail as select from vbak { vbeln, kunnr,
netwr, erdat }
```

✅ **Replaces:**

- Manual XML annotation, UI5 facet definition
  💼 **Use Case:** Fiori Object Page ready instantly.

📜 11. Field Label + Hidden Fields

```
@EndUserText.label: 'Customer Master'
@UI.lineItem: [{ position: 10, label: 'Customer Name' }]
@UI.hidden: true
define view entity ZI_CustHidden as select from kna1 { kunnr, name1, loevm
}
```

✅ **Replaces:**

- ALV catalog hiding logic
  💼 **Use Case:** Hide technical flags in Fiori apps.

🧩 12. Default Sort and Presentation Variant

```
@UI.presentationVariant: [{ sortOrder: [{ by: 'netwr', direction: #DESC }]
}]
@UI.chart: [{ type: #BAR, title: 'Top Customers', dimensions: ['kunnr'],
measures: ['netwr'] }]
define view entity ZI_TopCustChart as
  select from vbak
  group by kunnr
  { kunnr, sum( netwr ) as netwr }
```

✅ **Replaces:**

- ALV sorting + OData chart configuration
  💼 **Use Case:** “Top Customers by Sales” report.

🌐 E. Data Semantics & Localization Recipes

🌎 13. Auto Text Table Join + Language Handling

```
@ObjectModel.text.association: '_text'
define view entity ZI_Company as
  select from t001
    association [1..1] to t001t as _text
      on $projection.bukrs = _text.bukrs and _text.spras =
$session.system_language
  { bukrs, _text.butxt as company_name }
```

✅ **Replaces:**

- SELECT SINGLE from text tables
  💼 **Use Case:** Multi-language UI reporting.

💱 14. Currency and Unit Conversion

```
@Semantics.amount.currencyCode: 'waers'
@Semantics.quantity.unitOfMeasure: 'meins'
@EndUserText.label: 'Material Valuation'
define view entity ZI_Valuation as select from mseg { matnr, menge, meins,
dmbtr, waers }
```

✅ **Replaces:**

- Currency/unit formatting in reports
  💼 **Use Case:** Stock valuation display.

🕑 15. Business Date / Validity Period Handling

```
@Semantics.businessDate.validFrom: true
@Semantics.businessDate.validTo: true
define view entity ZI_PricingValid as select from a004 { matnr, datbi,
datab, kbetr }
```

✅ **Replaces:**

- Manual date range checks in loops
  💼 **Use Case:** Pricing condition validity display.

📋 F. Consumption & Value Help Recipes

🔍 16. Fiori Filter with Value Help

```
@Consumption.filter: true
@Consumption.valueHelpDefinition: [{ entity: { name: 'I_Material', element:
'Material' } }]
@UI.selectionField: [{ position: 10 }]
define view entity ZC_MaterialFilter as select from mara { matnr, mtart }
```

✅ **Replaces:**

- Custom SELECT-OPTIONS + F4 help coding
  💼 **Use Case:** Fiori search filter dropdown.

🔎 17. Default Search Element

```
@Search.searchable: true
@Search.defaultSearchElement: true
define view entity ZI_SearchCust as select from kna1 { kunnr, name1, ort01
}
```

✅ **Replaces:**

- Custom search box implementation
  💼 **Use Case:** Enable live search in Fiori lists.

🔍 G. Extensibility & Maintenance Recipes

🧩 18. Enable CDS View Extensibility

```
@Metadata.allowExtensions: true
@EndUserText.label: 'Extendable Material View'
define view entity ZI_Material as select from mara { matnr, mtart }
```

✅ **Replaces:**

- Implicit enhancements in SE11
  💼 **Use Case:** Allow partners/customers to extend views.

🧱 19. CDS View Extension Example

```
extend view ZI_Material with ZI_Material_Ext { zzcategory, zzsource }
```

✅ **Replaces:**

- APPEND structure approach in DDIC
  💼 **Use Case:** Add custom fields to released views safely.

🧮 20. Inline Calculations + Labeling

```
@EndUserText.label: 'Discounted Sales'
@UI.lineItem: [{ position: 10, label: 'Net Sales' }, { position: 20, label:
'Discounted Value' }]
define view entity ZI_Discount as
  select from vbak
  { vbeln, netwr, netwr * 0.95 as discounted_value }
```

✅ **Replaces:**

- Internal table field computation in loops
  💼 **Use Case:** Calculated field display in Fiori.

🧠 Final Takeaway — ABAP SME Insight

In classic ABAP, you'd write **dozens of lines of code** for:

- `AUTHORITY-CHECK`,
- ALV field catalogs,
- search helps,
- UI metadata,
- OData models,
- KPI dashboards,
- and text joins.

With these annotation "recipes", you **declare once and reuse forever** — no SE80, no dynpro, no loops.
This is how **modern ABAP on HANA** replaces workarounds with **semantic, declarative design**.
