🔷 Realtime CDS View Requirements & Detailed Solutions

#	Requirement (Scenario)	Detailed CDS View Solution (Step-by-Step)
1	Create real-time Fiori analytical report on Sales Orders with Gross Margin	Create a CDS with `@Analytics.query: true`. Join VBAK (header), VBAP (items), and KONV (pricing). Define calculated fields like `NetValue - Cost` for margin. Use `@Semantics.amount.currencyCode` for currency handling.
2	Combine data from multiple tables like VBAK, KNA1, and MARA with semantic layers	Build *basic* CDS views for each table, then create a *composite view* joining them using associations. Finally, expose through a *consumption view* with annotations for Fiori consumption. This ensures modularity and reuse.
3	Expose Purchase Order data to Fiori app or API	Add `@OData.publish: true` annotation to your CDS. Activate it; the system automatically generates OData service. Consume via `/n/IWFND/MAINT_SERVICE` and bind in Fiori app.
4	Restrict report data to user's Sales Org	Create a DCL (Data Control Language) object linked to the CDS view. Define authorization check like: `@MappingRole: 'ZROLE_SALESORG'`. Automatically filters data per logged-in user's authorization.
5	Enable drilldown from header to item in Fiori report	Create header CDS and associate items via `_Items` association. Use `@ObjectModel.text.association` for descriptive text. The Fiori Smart Template reads the hierarchy automatically.
6	Get customer or material text dynamically (lazy load)	Instead of a full join, use an *association* to text tables (e.g., `KNA1` → `_CustomerText`). Fiori loads text only when needed, improving performance.
7	Use one CDS for both analytics and transactional reporting	Mark data model views as `@Analytics.dataCategory: #CUBE` or `#DIMENSION`. The CUBE can feed analytics, while the same view can serve RAP apps through OData exposure.
8	Optimize performance of heavy CDS joins	Move join logic into associations and restrict cardinality. Avoid unnecessary joins by filtering early (`WHERE` clause). Use indexing hints and `@AbapCatalog.buffering: #OFF` for real-time data.
9	Display currency/unit conversions	Add semantic annotations: `@Semantics.amount.currencyCode: 'CurrencyField'` and `@Semantics.quantity.unitOfMeasure: 'UnitField'`. The UI framework automatically converts values.
10	Provide F4 Help for Material field	Create a dedicated *Value Help CDS* (VH view). Example: `ZVH_MATNR` selecting MATNR and description. Annotate your main CDS field with `@Consumption.valueHelpDefinition: [{entity: 'ZVH_MATNR'}]`.
11	Add calculated field like "Delivery Delay"	Define a virtual field: `cast( actual_delv_date - planned_delv_date as abap.int4 ) as DeliveryDelay`. You can also use `CASE` to handle null dates.

#	Requirement (Scenario)	Detailed CDS View Solution (Step-by-Step)
12	Handle multilingual text display	Create a separate text CDS on text tables (like `MAKT`). Use `@ObjectModel.text.association: '_Text'` and `@ObjectModel.text.element: ['Description']` to dynamically pull texts by user language.
13	Expose data to BW or SAC	Set `@Analytics.dataCategory: #CUBE` and `@VDM.viewType: #COMPOSITE`. Use `@AnalyticsDetails.query: true`. The BW extractor can directly consume the CDS as a data source.
14	Build open item aging report	Use date functions: `days_between( docdate, current_date )` to calculate aging. Group data by customer or document type using `GROUP BY`.
15	Filter report by parameters (Plant, Fiscal Year)	Add CDS parameters like `@Environment.systemField: #CLIENT` and `@Consumption.filter: true` for fields. CDS supports dynamic filters in Fiori automatically.
16	Derive logic for “High Value Customer”	Add a CASE expression inside SELECT list: `CASE WHEN total_value > 100000 THEN 'Y' ELSE 'N' END AS HighValue`. This logic executes in HANA, not ABAP.
17	Support grouping, sorting, and subtotaling in UI	Use `GROUP BY` in the CDS and annotate fields with `@AnalyticsDetails.query.axis: #ROWS` or `#COLUMNS` to help Fiori Smart Table understand layout.
18	Combine SAP and external table data	Use table aliases with schema-qualified names. Example: `FROM ZSALES AS S INNER JOIN "EXTSCHEMA"."EXT_TABLE" AS E.` Ensure external schema is whitelisted in `DBCON`.
19	Fetch latest record per document	Use `ROW_NUMBER()` window function: `row_number() over ( partition by docno order by changedate desc ) as rn` then filter `WHERE rn = 1`.
20	Real-time join with external HANA table	Use `@AbapCatalog.buffering: #OFF` to avoid buffer delays. Ensure data type alignment between CDS and external HANA structure.
21	Multilingual hierarchy display	Build text CDS for hierarchy nodes and associate using `@ObjectModel.text.association`. UI automatically switches texts by language.
22	Build organizational hierarchy (parent-child)	Use recursive CDS hierarchy definition: `@ObjectModel.hierarchy.association: '_Parent'` with parent-child relationship fields.
23	Restrict fields in report for specific users	Create *restricted consumption view* selecting only allowed fields. Alternatively, use CDS extension with `EXTEND VIEW` for role-specific layouts.
24	Build KPI Tile for CDS	Add `@Analytics.query: true` and expose the CDS as KPI in Smart Business Cockpit (SAP Fiori). Configure KPIs via Fiori Launchpad Designer.
25	Debug CDS data flow	Check generated SQL via `ST05` or `DBACOCKPIT`. Validate view data via `SE11 → Contents`. Use `EXPLAIN PLAN` to identify bottlenecks.

🔷 Realtime AMDP Requirements & Detailed Solutions

#	Requirement (Scenario)	Detailed AMDP Solution (Step-by-Step)
1	Handle large data aggregation efficiently	Create AMDP class implementing `IF_AMDP_MARKER_HDB`. Inside method, write SQLScript with `GROUP BY` and CE functions. Return results as internal table type to ABAP.
2	Replace nested SELECTs/loops	Consolidate logic into one SQLScript query. Example: Use joins and aggregations directly in AMDP, removing redundant ABAP loops.
3	Apply tier-based discount rule engine	Use SQLScript `CASE` expressions with thresholds. Pass thresholds as parameters to AMDP method.
4	Perform matrix pivot (e.g., region vs. product sales)	Use SQLScript pivot logic: `SELECT region, SUM(CASE WHEN product = 'A' THEN qty ELSE 0 END)`. Handle dynamically with `EXECUTE IMMEDIATE`.
5	Do mass update of transactional data	Inside AMDP, write `UPDATE ZTABLE SET FIELD = ... WHERE condition`. Trigger via ABAP report; reduces DB roundtrips drastically.
6	Retrieve top N per customer	Use analytic function: `RANK() OVER (PARTITION BY customer ORDER BY sales DESC)` and filter `WHERE rank <= :iv_top`.
7	Extend CDS logic with custom SQLScript	Create *table function* returning structure. Implement using AMDP method and consume via CDS `@AbapCatalog.sqlViewAppendName`.
8	Execute predictive model (PAL API)	Inside AMDP, call predictive function `APPLY_PREDICTIVE_MODEL('model_name', :input_data)`. Pass training data via parameter table.
9	Delta load for ETL	Use last extraction timestamp (`lt_last_ts`) and add `WHERE updated_on > :lt_last_ts`. Return delta dataset only.
10	Join SAP and external schema tables	Declare `USING SCHEMA_NAME.TABLE_NAME` in AMDP. Maintain schema mapping in DBCON for secure access.
11	Create temporary table for staging	Use `CREATE LOCAL TEMPORARY TABLE #temp` then perform intermediate calculations. Drop automatically after execution.
12	Handle recursive BOM explosion	Use `WITH RECURSIVE` CTE to process parent-child recursively until no child exists. Return flattened hierarchy table.
13	Currency conversion at DB layer	Join currency table (TCURR) within AMDP. Multiply values with rate fetched via latest valid date filter.
14	Parallelize data aggregation	Split data into partitions using `MOD(hash_field, N)` and aggregate in parallel inside AMDP using table variables.
15	Dynamic WHERE clause	Use `EXECUTE IMMEDIATE` to build query string with variable filters and execute dynamically.
16	Complex aggregation beyond CDS	Implement *AMDP Table Function* returning complex aggregations (e.g., weighted average) and consume via CDS.
17	Replace ABAP job summarization	Move summarization job to AMDP logic. Trigger through background program for performance gain (up to 90%).

#	Requirement (Scenario)	Detailed AMDP Solution (Step-by-Step)
18	Integrate with RAP behavior action	Define AMDP Table Function for heavy logic. Expose via RAP action handler method calling AMDP.
19	Validate before posting	Write AMDP validation returning error table. Check results before calling posting BAPI in ABAP layer.
20	Write audit log of each execution	At end of AMDP, insert into audit table using `INSERT INTO ZAUDIT_LOG VALUES (...).`
21	Real-time reconciliation report	Fetch latest data from multiple transactional tables with timestamp filters inside AMDP and return summary.
22	Multi-join performance optimization	Use CE functions (`CE_JOIN`, `CE_PROJECTION`) or common table expressions (CTEs) to optimize join paths.
23	Complex date interval logic	Use HANA SQL functions like `ADD_DAYS`, `MONTHS_BETWEEN`, `DAYNAME` to handle fiscal interval logic.
24	Fallback lookup logic	Use `COALESCE(value, default_value)` or CASE WHEN NULL conditions to provide fallback defaults.
25	Capture performance metrics	Use `EXPLAIN PLAN FOR` or `PLAN_TABLE` insertions inside AMDP to record query runtime and cost metrics.

✅ **How to Use This Reference:**

- For *each project requirement*, identify if the logic can be handled at the **CDS layer (declarative)** or needs **SQLScript (AMDP)**.
- Always prefer CDS first (simpler, reusable, UI-friendly), and move to AMDP when logic is **iterative, recursive, or highly computational**.
- Never paste production code into AI tools — instead, describe the logic in *pseudocode or functional terms* for safe assistance.
