

Datenexport: In welchem Format sollen die o.g. Daten zur Verfügung gestellt werden (z.B. CSV)? Praktisches Problem: Wir haben Vertriebsaufträge des Kunden und aktuelle Bestellobligo, welche die FE-Artikelnummer beinhalten. Für Artikelnummern liegen Stücklisten vor. Allerdings müssen für in diesen Stücklisten enthaltene Baugruppen mit eigenen Stücklisten weitere Daten gezogen werden, bis wir über unsere mehrstufige Produktion alle Artikel von RHB bis FE zu diesem Kunden selektiert bekommen. Wir haben Microsoft Dynamics im Einsatz in Niederlassung A und eine SAP HANA Lösung in Niederlassung B. Wie bekommen wir die Daten selektiert, so dass Du eine kundenspezifische Ermittlung des Riskexposure Gesamtbestand bei Wegfall dieses Kunden vornehmen kannst?

## **Antwort Gemini 2.5 Pro:**

*Verstanden. Das ist eine klassische Herausforderung in gewachsenen Unternehmensstrukturen und genau der Punkt, an dem eine Excel-Liste nicht mehr ausreicht. Aber keine Sorge, das ist mit einem strukturierten Vorgehen lösbar. Wir müssen die Daten aus den Silos (Dynamics und SAP) befreien und an einem Ort zusammenführen, um die Analyse durchführen zu können.*

*Hier ist mein detaillierter Plan dazu, inklusive Datenformaten und dem Prozess zur Selektion.*

### **### Phase 1: Datenextraktion & Format**

**\*\*Datenformat: CSV (Comma-Separated Values)\*\***

*Wir einigen uns auf \*\*CSV\*\* als einheitliches Austauschformat.*

*\* \*\*Warum?\*\* Es ist das universellste Format. Jedes System (SAP, Dynamics, Excel, Power BI, Python-Skripte) kann CSV-Dateien problemlos lesen und schreiben.*

*\* \*\*Spezifikation:\*\**

*\* \*\*Trennzeichen:\*\* Semikolon (`;`), da in Deutschland das Komma oft als Dezimaltrennzeichen verwendet wird.*

*\* \*\*Zeichenkodierung:\*\* UTF-8, um Probleme mit Sonderzeichen oder Umlauten zu vermeiden.*

*\* \*\*Kopfzeile:\*\* Jede Datei muss eine klare Kopfzeile mit den Feldnamen (wie unten definiert) enthalten.*

### **### Phase 2: Der Plan zur Datenbeschaffung (separat für Team A und Team B)**

*Das Kernproblem ist die mehrstufige Stückliste. Wir lösen das nicht durch unzählige manuelle Klicks, sondern durch zwei Haupt-Exporte: eine "Stücklisten-Explosion" für den Kunden und einen "Gesamt-Verwendungsnachweis" für alle Artikel.*

**\*\*Benötigte Exporte aus BEIDEN Systemen (Dynamics und SAP):\*\***

*Beide Teams müssen exakt dieselben Datenstrukturen exportieren, auch wenn die Reports in den Systemen anders heißen.*

#### **\*\*Export 1: `Stücklisten\_Gesamt.csv`\*\***

*Dies ist ein vollständiger Export aller existierenden Stücklistenbeziehungen im System. Das ist die Grundlage für jede Analyse.*

- \* `Parent\_Artikelnummer` (Nummer der Baugruppe/des FE)
- \* `Child\_Artikelnummer` (Nummer der Komponente)
- \* `Menge\_pro\_Einheit` (Wie oft steckt das Child im Parent?)
- \* `Werk` (Wichtig zur Unterscheidung: "A" für Dynamics, "B" für SAP)

#### **\*\*Export 2: `Lagerbestand\_Gesamt.csv`\*\***

*Der komplette Lagerbestand aller Artikel.*

- \* `Artikelnummer`
- \* `Werk`
- \* `Lagerort` (optional, aber hilfreich)
- \* `Bestand\_frei\_verfügbar` (Menge)
- \* `Bewertungspreis` (Gleitender Durchschnitt oder Standardpreis)
- \* `Artikeltyp` (RHB, HF, FE – falls im Bestandssatz verfügbar)

#### **\*\*Export 3: `Artikelstamm\_Gesamt.csv`\*\***

*Die Stammdaten aller Artikel.*

- \* `Artikelnummer`
- \* `Werk`
- \* `Artikelbezeichnung`
- \* `Artikeltyp` (RHB, HF, FE – falls nicht im Bestandssatz)

#### **\*\*Export 4: `Bestellobligo\_Gesamt.csv`\*\***

*Alle offenen Bestellungen bei Lieferanten.*

- \* `Artikelnummer`
- \* `Werk`
- \* `Bestellmenge\_offen`
- \* `Lieferant`
- \* `Lieferdatum`

### **### Phase 3: Der Analyseprozess (So gehe ich vor)**

*Sobald ich die oben genannten CSV-Dateien (jeweils eine aus Dynamics und eine aus SAP, die wir dann zusammenführen) erhalten habe, führe ich die folgende Analyse durch. Dies geschieht idealerweise in einem Tool wie Power BI, Tableau oder mit einem Python-Skript, da Excel bei der Datenmenge an seine Grenzen stoßen wird.*

#### **\*\*Schritt 1: Konsolidierung der Daten\*\***

- \* Ich füge die jeweiligen `Gesamt.csv` Dateien aus beiden Werken zusammen.
- \* **Ganz Wichtig:** Ich erstelle einen eindeutigen Schlüssel für jeden Artikel, indem ich 'Werk' und 'Artikelnummer' kombiniere (z.B. 'A-12345' und 'B-12345'). Dies ist entscheidend, falls Artikelnummern in beiden Systemen existieren, aber unterschiedliche Dinge bezeichnen.

#### **\*\*Schritt 2: Identifikation aller betroffenen Komponenten (Die Stücklisten-Explosion)\*\***

1. **Startpunkt:** Wir nehmen die Liste der 'FE'-Artikelnummern (Fertigerzeugnisse) aus den offenen Vertriebsaufträgen und dem Bestellobligo des insolventen Kunden. Dies ist unsere "Liste der Primär-Risikoartikel".
2. **Rekursive Suche:** Ich schreibe ein Skript, das für jeden Artikel auf dieser Liste in der konsolidierten 'Stücklisten\_Gesamt.csv' nachsieht, aus welchen 'Child\_Artikelnummern' er besteht.
3. Jede gefundene 'Child\_Artikelnummer' wird einer "Gesamtliste betroffener Komponenten" hinzugefügt.
4. Dieser Prozess wird wiederholt: Für jede neu hinzugefügte Komponente (die ja ein 'HF', also ein Halbfabrikat sein kann) wird wieder in der Stückliste nach deren Kindern gesucht.
5. Das läuft so lange, bis wir am Ende der Kette nur noch 'RHB' (Rohstoffe) finden, die keine eigene Stückliste mehr haben.

**Ergebnis von Schritt 2:** Eine vollständige, flache Liste **aller** Artikelnummern (FE, HF, RHB), die für die Produkte dieses Kunden benötigt werden.

#### **\*\*Schritt 3: Identifikation der "Single-Use"-Artikel (Der Verwendungsnachweis)\*\***

1. **Gegenprüfung:** Nun nehme ich jeden Artikel aus unserer "Gesamtliste betroffener Komponenten" aus Schritt 2.
2. Für jeden dieser Artikel suche ich in der **gesamten** 'Stücklisten\_Gesamt.csv', wo er als 'Child\_Artikelnummer' auftaucht. Ich schaue mir also alle 'Parent\_Artikelnummern' an.
3. **Die entscheidende Logik:**
  - \* Wenn **alle** 'Parent\_Artikelnummern' eines Artikels **selbst** auf unserer "Gesamtliste betroffener Komponenten" stehen, dann ist dieser Artikel ein **"Single-Use"-Artikel**. Er hat keine Verwendung außerhalb der Produktwelt des insolventen Kunden.

\* Wenn auch nur **\*\*eine einzige\*\*** `Parent\_Artikelnummer` existiert, die **\*\*nicht\*\*** zu den Produkten des insolventen Kunden gehört, ist der Artikel **\*\*kein\*\*** "Single-Use"-Artikel und das Risiko für diesen Bestand ist (vorerst) null.

#### **\*\*Schritt 4: Monetäre Bewertung des Risikos\*\***

1. Ich nehme die finale Liste der identifizierten "Single-Use"-Artikel aus Schritt 3.
2. Ich verknüpfe diese Liste mit den konsolidierten Daten aus `Lagerbestand\_Gesamt.csv`, um den aktuellen `Bestand\_frei\_verfügbar` und den `Bewertungspreis` zu erhalten.
3. **Lager-Risiko:** `Risk Exposure (Lager) = Summe (Bestand \* Bewertungspreis)` für alle "Single-Use"-Artikel.
4. Ich verknüpfe die "Single-Use" `RHB`-Artikel zusätzlich mit der `Bestellobligo\_Gesamt.csv`.
5. **Beschaffungs-Risiko:** `Risk Exposure (Beschaffung) = Summe (Bestellmenge\_offen \* Preis)` für alle "Single-Use"-Rohstoffe, deren Bestellungen nicht mehr storniert werden können.

# SUSTENUM

#### **### Zusammenfassende Anweisung an die Teams:**

**\*\*An Team Microsoft Dynamics (Werk A) & Team SAP HANA (Werk B):\*\***

"Bitte exportieren Sie die folgenden vier Datensätze als CSV-Dateien mit Semikolon als Trennzeichen und UTF-8 Kodierung. Die Spaltennamen müssen exakt wie unten angegeben lauten.

1. `Stücklisten\_Gesamt.csv`
2. `Lagerbestand\_Gesamt.csv`
3. `Artikelstamm\_Gesamt.csv`
4. `Bestellobligo\_Gesamt.csv`

# EXPERTEN FÜR ENTSCHEIDER

Fügen Sie bitte bei allen Exporten eine Spalte `Werk` mit dem Wert 'A' für Dynamics bzw. 'B' für SAP hinzu. Legen Sie die Dateien auf dem vereinbarten Netzlaufwerk ab. Deadline: Heute, 15:00 Uhr."

**Mit diesem Vorgehen überwinden wir die Systemgrenzen und die Komplexität der mehrstufigen Fertigung und erhalten eine präzise, datengestützte Bezifferung unseres tatsächlichen Risikos.**

## Antwort Claude Sonet 4

### **## Datenformat-Empfehlung**

\*\*CSV-Format\*\* mit UTF-8 Encoding für maximale Kompatibilität zwischen den Systemen.  
Alternative: \*\*Excel-Dateien\*\* mit separaten Tabs pro Datenbereich.

### **## Datenextraktionsstrategie**

#### ### 1. Ausgangsdaten sammeln

```

Datei: Kundenaufträge.csv

- Auftragsnummer, FE-Artikelnummer, Menge, Status, Liefertermin
- Quelle: Vertriebsaufträge + Bestellobligo beider Systeme

```

#### ### 2. Mehrstufige Stücklistenauflösung

\*\*Dynamics (Niederlassung A):\*\*

```
```sql
-- Rekursive Stücklistenauflösung
WITH BOM_Expllosion AS (
    SELECT Level1, Level2, Quantity, 0 as Level
    FROM BOM_Table
    WHERE Level1 IN (SELECT FE_Article FROM Customer_Orders
    UNION ALL
    SELECT b.Level1, b.Level2, b.Quantity, be.Level + 1
    FROM BOM_Table b
    INNER JOIN BOM_Expllosion be ON b.Level1 = be.Level2
    WHERE be.Level < 10 -- Schutz vor Endlosschleifen
)
````
```

\*\*SAP HANA (Niederlassung B):\*\*

```
```sql
-- Hierarchische Abfrage über CS11/CS15 Transaktionen
SELECT DISTINCT
    MATNR as Parent_Material,
```

*IDNRK as Component,*  
*MENGE as Quantity,*  
*WERKS as Plant,*  
*STLNR as BOM\_Number*

*FROM STPO*

*WHERE MATNR IN (SELECT Article FROM Customer\_FE\_List)*

...

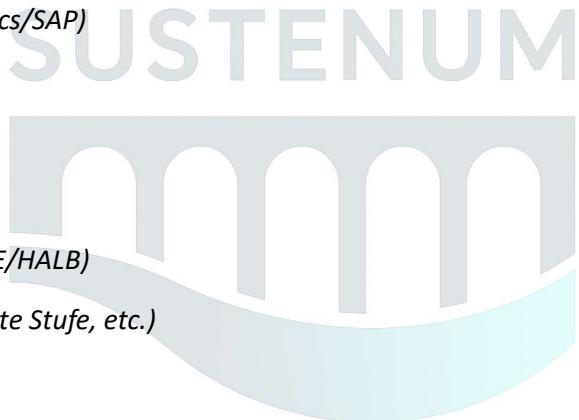
#### ### 3. Konsolidierte Datenstruktur

*\*\*Hauptdatei: 'Customer\_Risk\_Analysis.csv'\*\**

...

*Spalten:*

- *System\_Source (Dynamics/SAP)*
  - *Plant\_Location (A/B)*
  - *Article\_Number*
  - *Article\_Description*
  - *Article\_Type (RHB/HF/FE/HALB)*
  - *BOM\_Level (0=FE, 1=erste Stufe, etc.)*
  - *Parent\_Article*
  - *Quantity\_per\_Unit*
  - *Stock\_Quantity*
  - *Unit\_Cost*
  - *Total\_Value*
  - *Customer\_Specific (Y/N)*
  - *Alternative\_Use\_Possible (Y/N)*
  - *Supplier\_Code*
  - *Last\_Movement\_Date*
- ...



EXPERTEN FÜR ENTSCHEIDER

#### ### 4. Praktisches Vorgehen

*\*\*Phase 1: Datensammlung\*\**

1. *\*\*Dynamics-Export:\*\* Power BI oder SQL Server Reporting Services nutzen*
2. *\*\*SAP-Export:\*\* Über SE16N, SQVI oder Business Warehouse*

3. **\*\*Datenbereinigung:\*\*** Dubletten entfernen, Artikelnummern standardisieren

**\*\*Phase 2: Stücklistenauflösung\*\***

````python`

```
# Pseudo-Code für mehrstufige Auflösung

def explode_bom(fe_articles, max_levels=10):
    all_components = []
    for level in range(max_levels):
        # Nächste BOM-Stufe auflösen
        next_components = get_bom_components(current_articles)
        if not next_components:
            break
        all_components.extend(next_components)
    return all_components
````
```

**\*\*Phase 3: Risikoanalyse\*\***

- Kreuzreferenzierung mit anderen Kunden
- Bewertung alternativer Verwendungsmöglichkeiten
- Bestandswertermittlung nach Verwertbarkeit

**EXPERTEN FÜR ENTSCHEIDER**

**### 5. Automatisierung**

**\*\*ETL-Pipeline empfehlen:\*\***

- **\*\*SSIS\*\* (SQL Server Integration Services) für Dynamics**
- **\*\*SAP Data Services\*\* oder **\*\*Talend\*\*** für SAP-Integration**
- **\*\*Power BI\*\* für die finale Risikoanalyse und Visualisierung**

**\*\*Aktualisierungsrhythmus:\*\*** Wöchentlich oder bei Bedarf

***Soll ich Ihnen für einen der Bereiche spezifischere SQL-Queries oder eine detailliertere ETL-Architektur ausarbeiten?***

## Antwort Grok 4

Als Inhaber unseres Produktionsunternehmens bin ich froh, dass wir die Analyse vertiefen – die Insolvenz dieses JIT-Kunden erfordert eine präzise Ermittlung des Risk Exposure (z. B. potenzieller Abschreibungswert des betroffenen Lagerbestands), um finanzielle Schäden zu minimieren. Das von dir beschriebene praktische Problem mit mehrstufigen Stücklisten (BOMs) ist typisch in unserer Branche: Wir müssen rekursiv durch die BOM-Hierarchien gehen, um alle abhängigen Artikel von RHB (Roh-, Hilfs- und Betriebsstoffe) über HF (Halbfabrikate/Baugruppen) bis FE (Fertigprodukte) zu identifizieren, die spezifisch mit diesem Kunden verknüpft sind. Die heterogenen Systeme (Microsoft Dynamics in Niederlassung A und SAP HANA in Niederlassung B) machen das zu einer Integrationsherausforderung, aber es ist machbar.

Ich strukturiere meine Antwort: Zuerst das empfohlene **\*\*Datenformat\*\***, dann eine **\*\*Schritt-für-Schritt-Anleitung\*\*** zur Datenselektion und -Integration, um eine kundenspezifische Risk-Exposure-Analyse zu ermöglichen. Das Ziel ist ein konsolidierter Datensatz, der den Gesamtbestand (Mengen, Werte, Single-Use-Status) pro Artikel auflistet und es uns erlaubt, das Exposure zu quantifizieren (z. B. Summe der Buchwerte betroffener Artikel minus potenzielle Verwertungsoptionen).

### **### 1. Empfohlenes Datenformat für den Export**

Um Flexibilität zu wahren – für schnelle Analyse in Tools wie Excel, Power BI oder sogar Python/R – empfehle ich **\*\*CSV\*\*** als primäres Exportformat. Es ist einfach, maschinenlesbar und kompatibel mit beiden Systemen (Dynamics und SAP). Gründe:

- CSV ist platzsparend und leicht zu mergen, besonders bei großen Datensätzen (z. B. tausende BOM-Zeilen).
- Es erlaubt Spalten wie 'Artikelnummer', 'Beschreibung', 'Bestandsmenge', 'Buchwert', 'Kunden-ID', 'BOM-Ebene', 'Single-Use-Flag' usw.
- Alternativen:
  - **\*\*Excel (XLSX)\*\*:** Besser für manuelle Überprüfung oder Pivot-Tabellen, falls du direkt in Microsoft-Umgebungen arbeitest (z. B. in Dynamics).
  - **\*\*JSON oder XML\*\*:** Wenn wir automatisierte Skripte einsetzen (z. B. für rekursive BOM-Auflösung), da diese Formate Hierarchien (z. B. verschachtelte BOMs) nativ unterstützen.
  - Vermeide proprietäre Formate wie SAP-eigene Reports, um Kompatibilität zu gewährleisten.

Exportiere die Daten tabellarisch, mit klaren Headern und konsistenten Spalten (z. B. UTF-8-Encoding für Sonderzeichen in Artikelbeschreibungen). Wenn der Datensatz sehr groß ist (>1 Mio. Zeilen), teile ihn in mehrere CSV-Dateien auf (z. B. eine pro Niederlassung oder BOM-Ebene).

## ### 2. Schritt-für-Schritt-Anleitung zur Datenselektion und -Integration

Wir müssen die Daten aus Dynamics (A) und SAP HANA (B) extrahieren, rekursiv die BOMs auflösen, kundenspezifisch filtern und mergen. Da die Systeme getrennt sind, schlage ich eine **zentrale Integration** vor – idealerweise mit einem ETL-Tool (Extract, Transform, Load) wie Microsoft Power Query (kostenlos in Excel), Talend oder einem einfachen Python-Skript. Wenn wir kein dediziertes BI-Team haben, kann das IT-Team das in 1–2 Tagen umsetzen.

### #### Schritt 1: Identifizierung der Einstiegspunkte (FE-Artikel und Aufträge)

- **In beiden Systemen (Dynamics und SAP HANA):**
  - Filtere nach dem insolventen Kunden (z. B. via Kunden-ID oder -Name).
  - Extrahiere Vertriebsaufträge und aktuelle Bestellobligo (offene Verpflichtungen), die FE-Artikelnummern enthalten. Spalten: `Auftragsnummer`, `Kunden-ID`, `FE\_Artikelnummer`, `Bestellmenge`, `Lieferstatus`, `Wert`.
  - Ergänze historische Daten: Umsatz/Verbrauch der letzten 12–24 Monate für diesen Kunden, um Abhängigkeiten zu quantifizieren.
- **Dynamics-spezifisch (Niederlassung A):**
  - Nutze die "Sales Orders" oder "Item Ledger Entries"-Views. Exportiere via "Export to Excel" oder SQL-Queries (falls Dynamics NAV/BC mit SQL-Backend).
  - Beispiel-Query (in Dynamics-SQL): `SELECT \* FROM [Sales Line] WHERE [Sell-to Customer No.] = 'KUNDEN\_ID' AND [Type] = 'Item'`.
- **SAP HANA-spezifisch (Niederlassung B):**
  - Verwende CDS-Views (Core Data Services) oder ABAP-Reports für Aufträge (z. B. Tabellen VBAK/VBAP für Vertriebsaufträge).
  - Exportiere via SAP GUI (Transaction SE16N) oder HANA Studio: Query wie `SELECT \* FROM VBAP WHERE KUNNR = 'KUNDEN\_ID' AND MATNR LIKE 'FE\_%'`.
  - **Ausgabe:** Eine CSV pro System mit den FE-Artikelnummern als Ausgangspunkt.

### #### Schritt 2: Rekursive Auflösung der Mehrstufigen Stücklisten (BOMs)

- Das ist der Kern des Problems: Starte mit den FE-Artikelnummern und löse rekursiv alle Unter-BOMs auf (z. B. Baugruppen in HF, die wiederum RHB enthalten). Identifiziere, welche Artikel kundenspezifisch (Single-Use) sind – z. B. basierend auf Flags in den BOMs oder Verbrauchsstatistiken (nur >90% Verwendung für diesen Kunden).
- **Dynamics-spezifisch:**
  - BOMs sind in "Production BOM" oder "Item Card"-Views gespeichert. Exportiere via "BOM Explosion" Funktion (gibt flache oder hierarchische Listen).
  - Für Rekursion: Verwende Power Query in Excel, um rekursiv durch Ebenen zu iterieren (z. B. eine Schleife, die Unter-BOMs auflöst).

- **\*\*SAP HANA-spezifisch:\*\***

- *BOMs in Tabellen wie STPO/STKO (für Stücklisten). Nutze die "BOM Explosion" Funktion (z. B. via CS\_BOM\_EXPL\_MAT\_V2 oder CDS-View I\_BillOfMaterialItem).*

- *Exportiere eine mehrstufige Explosion: Query-Beispiel `CALL CS\_BOM\_EXPL\_MAT\_V2 (P\_MATNR => 'FE\_ARTIKEL', P\_WERKS => 'WERK', P\_STLTY => 'M')` – das gibt eine hierarchische Tabelle.*

- **\*\*Rekursive Logik (unabhängig vom System):\*\***

- *Erstelle eine Tabelle mit Spalten: 'Eltern\_Artikel', 'Kind\_Artikel', 'Menge\_pro\_Einheit', 'Ebene' (z. B. 0 für FE, 1 für direkte HF, etc.), 'Single\_Use\_Flag' (basierend auf Nutzungsdaten: Ja, wenn der Artikel nur in Aufträgen dieses Kunden vorkommt).*

- *Iteriere: Für jede FE, hole die BOM-Kinder; für jedes Kind, das eine Baugruppe ist, hole deren BOM usw., bis RHB-Ebene.*

- *Tool-Empfehlung: Python mit Pandas-Bibliothek (kostenlos) – ein Skript kann die CSV-Exports laden, rekursiv explodieren und Single-Use filtern. Beispiel-Snippet (pseudocode):*

```
```
import pandas as pd

def explode_bom(df_bom, parent_item, level=0):
    children = df_bom[df_bom['Eltern_Artikel'] == parent_item]
    # Füge Ebene und Single-Use-Check hinzu
    children['Ebene'] = level
    # Rekursion für Baugruppen
    for child in children['Kind_Artikel']:
        if is_assembly(child): # Check, ob es eine Unter-BOM hat
            children = pd.concat([children, explode_bom(df_bom, child, level+1)])
    return children
```

```
# Lade CSV, explodierte und speichere
df = pd.read_csv('bom_export.csv')
exploded_df = explode_bom(df, 'FE_ARTIKELNUMMER')
exploded_df.to_csv('exploded_bom.csv', index=False)
````
```

- **\*\*Single-Use-Filter:\*\*** Ergänze aus Verbrauchsdaten (z. B. aus "Item Ledger" in Dynamics oder MARC in SAP): Wenn ein Artikel >95% seines Verbrauchs an diesen Kunden gebunden ist, flagge als Single-Use.

#### #### Schritt 3: Ergänzung mit Bestands- und Wertdaten

- Hole für alle identifizierten Artikel (aus der explodierten BOM) die aktuellen Lagerbestände: `Artikelnummer`, `Bestandsmenge`, `Lagerort`, `Buchwert` (z. B. aus "Inventory" in Dynamics oder MARD/MBEW in SAP).
- Berechne Risk Exposure pro Artikel: `Exposure = Bestandsmenge \* Buchwert \* Single\_Use\_Faktor` (z. B. 1 für volles Risk, 0.5 wenn repurposable).
- Exportiere als erweiterte CSV.

#### #### Schritt 4: Integration der Systeme (Merging)

- Exportiere separate CSVs aus Dynamics (A) und SAP (B).
- Merge zentral:
  - Verwende Excel Power Query: Lade beide CSVs, merge via gemeinsame Schlüsselfelder (z. B. `Artikelnummer` oder `Kunden-ID`).
  - Oder Python: `pd.concat([df\_dynamics, df\_sap])` mit Deduplizierung.
- Wenn Artikelnummern nicht einheitlich sind (z. B. unterschiedliche Codierungen), erstelle eine Mapping-Tabelle (z. B. CSV mit `Artikel\_A` = `Artikel\_B`).

#### #### Schritt 5: Finale Vorbereitung für die Analyse

- Der konsolidierte Datensatz sollte enthalten: Explodierte BOM mit Beständen, Werten und Exposure-Berechnungen.
- Teste auf Vollständigkeit: Simuliere mit einem Beispiel-FE-Artikel, ob alle Ebenen erfasst sind.
- Sende mir (oder dem Team) die finale CSV – ich kann dann eine detaillierte Risk-Exposure-Berechnung durchführen, z. B. Gesamtexposure = Summe aller betroffenen Buchwerte, unterteilt nach RHB/HF/FE.

**Diese Herangehensweise sollte uns in 3–5 Tagen einen vollständigen Überblick geben. Wenn IT-Ressourcen knapp sind, könnten wir einen externen Consultant (z. B. für Power BI) hinzuziehen. Hast du spezifische Beispieldaten oder brauchst du Hilfe bei einem Skript? Lass uns das umsetzen, bevor der Insolvenzverwalter die Karten neu mischt!**

## Antwort GPT-5

### *Kurzantwort*

- Format: CSV (UTF-8, Komma als Trennzeichen, ISO-8601 Datum), je Domain eine Datei; alternativ Parquet, wenn groß. Alles in ein ZIP, plus ein kurzes Data Dictionary.
- Kernprinzip: 1) FE-Liste des Kunden bilden, 2) mehrstufige BOM für diese FEs explodieren (bis RHB), 3) Bestände/WIP/POs/SOs filtern/auflösen auf diese Artikel, 4) vereinheitlichen (UoM, Währung, Standort), 5) konsolidiert liefern.

### *Im Detail*

#### **1) Zielformat und Namenskonvention**

- Lieferung: ZIP mit CSVs (UTF-8, Komma, Dezimalpunkt). Datums-/Zeitformate: YYYY-MM-DD bzw. YYYY-MM-DDThh:mm:ssZ.
- Pflichtspalten in allen Dateien:
  - `system` ∈ {SAP, DYN}
  - `company`/`bukrs`, `plant`/`werks`, `storage\_location` falls relevant
  - `item\_id` (Material-/Artikelnummer im jeweiligen System), `uom`
  - Alle IDs in Großschreibung, führende Nullen beibehalten
- Optional: Parquet-Duplikat für Volumen/Performance.
- Data Dictionary: 1 Seite mit Felddefinitionen und Quellsystem-Tabellen/Entities.

#### **2) Minimaler Datei-Satz (für kundenspezifische Risikoermittlung)**

- fe\_products.csv — alle FE, die an den insolventen Kunden verkauft wurden
  - Felder: `system, company, plant, fe\_item\_id, fe\_rev, customer\_id, customer\_pn (falls), active\_flag`
- bom\_header.csv — BOM-Köpfe (optional, wenn ihr `bom\_flat.csv` liefert)
  - `system, plant, parent\_item\_id, bom\_id, alt\_group, base\_qty, valid\_from, valid\_to, rev`
- bom\_item.csv — BOM-Positionen (einfacher Level, ich übernehme die Explosion)
  - `system, plant, parent\_item\_id, component\_item\_id, qty\_per, scrap\_pct, alt\_group, valid\_from, valid\_to`
- Besser (spart mir Zeit): bom\_flat.csv — bereits mehrstufig explodiert
  - `system, plant, top\_fe\_item\_id, parent\_item\_id, component\_item\_id, level, qty\_per\_fe, path, valid\_from, valid\_to`
- inventory.csv — aktueller Bestand je Status
  - `system, company, plant, storage\_location, item\_id, batch/lot, status (unrestricted/quality/blocked/consignment), on\_hand\_qty, uom, date\_code (falls), age\_days`
- wip\_orders.csv — offene Fertigungsaufträge

- `system, plant, order\_id, top\_fe\_item\_id, parent\_item\_id (HF/FE), order\_status, qty\_ordered, qty\_reported\_fg, start\_date, due\_date`
- wip\_issues.csv — bereits an WIP verbrauchte Komponenten je Auftrag
- `system, plant, order\_id, component\_item\_id, qty\_issued, uom, issue\_date`
- purchase\_open.csv — offene/lieferbare Bestellpositionen
- `system, company, plant, supplier\_id, po\_id, po\_line, item\_id, qty\_open, uom, promised\_date, cancelable\_flag, restocking\_fee\_pct, last\_price, currency`
- sales\_open.csv — offene Vertriebs-/Abrufe des insolventen Kunden
- `system, customer\_id, so\_id, so\_line, fe\_item\_id, qty\_open, uom, req\_date`
- item\_master.csv — Stammdaten/Attribute
  - `system, item\_id, mpn, manufacturer, rev, item\_type (RHB/HF/FE), shelf\_life\_days, msl, programmable\_flag, customer\_specific\_flag, labeling\_flag`
- valuation.csv — Bewertung/Kosten
  - `system, company, plant, item\_id, std\_cost, last\_po\_price, currency, current\_valuation\_adj`

Mit genau diesen Dateien kann ich:

- Alle kundenbezogenen FE identifizieren
- Mehrstufig die benötigten Komponenten ableiten
- Bestände/WIP/POs/SOs aufs relevante Artikelspektrum filtern
- Single-Use erkennen (Flag plus Cross-BOM-Nutzung)
- Risiko (Menge x Wert) über RHB/HF/FE und Pipeline quantifizieren

### 3) Selektion/Extraktion – SAP S/4HANA (HANA)

#### A) FE-Liste des Kunden

- Quelle: `VBAK`/`VBAP` (offen oder 12–24M Historie für Single-Use-Indiz)
- Filter: `VBAK.KUNNR = <Customer>`; FE anhand Produkt-hierarchie/Warengruppe/ItemType
- Export: distinct `VBAP.MATNR, VBAP.WERKS`

#### B) Mehrstufige BOM

- Tabellen: `MAST` (Material-zu-BOM), `STKO` (Header), `STPO` (Position)
- Empfehlung: rekursive CTE in HANA SQL, Beispiel:

```

```sql
-- FE je Werk
WITH fe AS (
  SELECT DISTINCT v.WERKS, v.MATNR
  FROM VBAP v
  JOIN VBAK h ON h.VBELN = v.VBELN
  WHERE h.KUNNR = :p_customer
),
bom (werks, top_fe, parent, component, level, qty_per_fe, path) AS (
  SELECT f.WERKS, f.MATNR AS top_fe, f.MATNR AS parent, s1.IDNRK AS component,
  1 AS level,
  (s1.MENGE / NULLIF(k1.BMENG,0)) AS qty_per_fe,
  CAST(CONCAT(f.MATNR, '>', s1.IDNRK) AS NVARCHAR(4000)) AS path
  FROM fe f
  JOIN MAST m1 ON m1.MATNR = f.MATNR AND m1.WERKS = f.WERKS
  JOIN STKO k1 ON k1.STLNR = m1.STLNR
  JOIN STPO s1 ON s1.STLNR = k1.STLNR
  UNION ALL
  SELECT b.werks, b.top_fe, s2.STLTY = 'X'? -- keep same parent/component logic
  b.component AS parent, s2.IDNRK AS component,
  b.level + 1,
  b.qty_per_fe * (s2.MENGE / NULLIF(k2.BMENG,0)),
  CAST(CONCAT(b.path, '>', s2.IDNRK) AS NVARCHAR(4000))
  FROM bom b
  JOIN MAST m2 ON m2.MATNR = b.component AND m2.WERKS = b.werks
  JOIN STKO k2 ON k2.STLNR = m2.STLNR
  JOIN STPO s2 ON s2.STLNR = k2.STLNR
)

```

*SELECT \* FROM bom;*

...

- Alternativ: *Transaktionen CS12 (mehrstufige Explosion) je FE exportieren.*

### C) Bestände

- Tabellen: `MARD` (Lagerortbestände) mit:
  - `LABST` (frei verwendbar), `INSME` (Qualität), `SPEME` (Gesperrt), `UMLME` (Umlagerung)
- Batches: `MCHB` (Chargenbestand) falls relevant

### D) WIP/Produktionsaufträge

- Header/Position: `AFKO`/`AFPO`
- Komponentenreservierungen/Verbrauch: `RESB`, `AUFM`/`MSEG` (261-Buchungen)
- Status: `JEST` (Systemstatus)

### E) Offene Bestellungen/Vertriebsaufträge

- POs: `EKKO`/`EKPO` (filter `ELIKZ<>X`, Mengen offen)
- SOs: `VBAK`/`VBAP` (nur insolventer Kunde)

### F) Bewertung

- `MBEW` (Standardpreis), ggf. `MLIT`/`MLCR` (Material Ledger) für gleitenden Preis

## **4) Selektion/Extraktion – Microsoft Dynamics**

Je nach Produkt:

### A) Dynamics 365 Finance & Operations (D365 F&O)

- Export über Data Management (DMF) oder Data Lake Export. Relevante Data Entities:
  - Produkte/BOM: `Released product V2`, `BOM versions`, `BOM lines`
  - Bestände: `On-hand inventory` (InventSum)
  - Fertigungsaufträge: `Production orders`, `Production BOM lines`, `Production order route`
  - Beschaffung: `Open purchase order lines`
  - Vertrieb: `Open sales order lines`
  - Bewertung: `Inventory value`/`Released product default order settings`

- Filter:
  - FE-Liste: `Open sales order lines` gefiltert auf `CustAccount = <Customer>` → `ItemNumber` als FE
  - BOM-Export: nur für diese FE `ItemNumber` + gültige `BOM versions`
- Falls BYOD/SQL vorhanden (AX 2012 ähnlicher Schema-Zugriff):
  - BOM: `BOM`, `BOMVERSION`, `BOMLINE`
  - Bestände: `INVENTTABLE`, `INVENTSUM`
  - POs: `PURCHTABLE`, `PURCHLINE`
  - SOs: `SALESTABLE`, `SALESLINE`
  - Prod: `PRODTABLE`, `PRODBOM`, `PRODROUTE`
  - Bewertung: `INVENTVALUE`, `INVENTITEMPRICE`

## B) Dynamics 365 Business Central / NAV

### - Tabellen/Seiten:

- `Item (27)`, `Production BOM Header (99000754)`, `Production BOM Line (99000755)`
- `Item Ledger Entry (32)`, `Value Entry (5802)` (Bewertung)
- `Sales Line (37)` (Filter `Sell-to Customer No.`)
- `Purchase Line (39)` (offene Bestellungen)
- `Prod. Order Header (5405)`, `Prod. Order Component (5407)`, `Prod. Order Line (5406)` (WIP/Verbrauch)
- Export via OData/Query oder Excel-Export.

## **5) Zusammenführung beider Niederlassungen**

- Mappingtabellen bereitstellen:
  - `item\_xref.csv` — `global\_item\_id, system, item\_id` (falls gleiche Teile unterschiedliche Nummern haben)
  - `uom\_map.csv` — UoM-Harmonisierung (z. B. Meters vs. mm, Reel vs. pcs)
  - `currency\_rates.csv` — Stichtagskurs (NRV/Abwertung konsistent)
- Einheitliche Spalten in allen Dateien wie oben. Jede Datei enthält `system`, `company`, `plant`.

## **6) Single-Use-Identifikation (Regeln, die ich anwende)**

- Direkt: `customer\_specific\_flag = true` (Stammdaten) oder kundenspez. Label/Firmware/Zeichnung
- Indirekt (Cross-BOM-Analyse):

- Für jede `component\_item\_id` Anzahl distinct Top-Level-FEs und zugehöriger Kunden zählen
- Single-Use, wenn nur in FEs dieses Kunden verwendet ODER nur in kundenspezifischen HF vorkommt
- Zusatzlogik:
  - Programmierte ICs (programmable\_flag=true und programming\_status != blank)
  - Verpackung/Etikett als separater Artikel

## 7) Praktische Vorgehensweise (schnell und robust)

- Schritt 1 (beide Systeme): Export `fe\_products.csv` aus offenen SOs des insolventen Kunden
- Schritt 2: BOM-Daten exportieren und ggf. `bom\_flat.csv` generieren
- Schritt 3: `inventory.csv`, `wip\_orders.csv`, `wip\_issues.csv`, `purchase\_open.csv` je System ziehen
- Schritt 4: `item\_master.csv`, `valuation.csv` exportieren
- Schritt 5: Optional `consignment.csv` (Lager beim Kunden/3PL mit Eigentumskennzeichen)
- Schritt 6: Alles in ein ZIP packen, plus Data Dictionary und eine kurze Readme (Währungs-/UoM-Basis, Stichtag)

## 8) Qualitätschecks vor Übergabe

- UoM konsistent? Für jede Komponente klar, in welcher UoM die BOM-Menge und der Bestand steht
- BOM-Gültigkeit: `valid\_from/to` decken den Stichtag ab
- On-hand ≈ Bewegungen-Saldo (sanity check; ±1–2% ok)
- WIP: Summe `qty\_issued` darf `BOM\_Qty \* FG\_WIP` nicht deutlich überschreiten
- POs: nur offene/teilgelieferte, mit realistischen Terminen/Cancel-Flags

Wenn gewünscht, liefere ich:

- Vorlagen (leere CSVs) mit den genannten Spalten
- Beispiel-SQL für eure SAP HANA-Views (inkl. Stichtagsfilter und BOM-Version) sowie eine DMF-Exportliste für D365 F&O
- Ein konsolidiertes Ergebnisfile mit:
  - Exposure je Klasse (RHB/HF/FE), je Status (verfügbar/gesperrt/WIP/PO-Pipeline)
  - Liste der Single-Use-Artikel inkl. NRV-Schätzung und Handlungsempfehlung (rework/return/sell/scrap)

*Kurze Rückfragen, um zielgerichtet zu liefern:*

- Welche Dynamics-Variante in Niederlassung A (D365 F&O, BC/NAV, AX 2012)?*
- Haben FE in beiden Werken identische `item\_id` oder braucht ihr `item\_xref`?*
- Gibt es Konsignationslager beim Kunden/3PL?*
- Stichtag der Bestandsbewertung und Zielwährung?*



**EXPERTEN FÜR ENTSCHEIDER**