Think BIGGER

DEEP DIVE FÜR FACHLEUTE

Microsoft Dynamics AX 2012: Architektur, Module, Datenbankschema und Migration

Technische Architektur, Datenbankschema, Entwicklung und Migrationspfad nach D365 FO

Autor	Erhard Rainer
Veröffentlicht	31.05.2024
Aktualisiert	07.07.2026
Umfang	ca. 30 Min. Lesezeit
Online	www.thinkbigger.at/wissen/dynamics-ax-2012/

thinkbigger.at

Business Intelligence · Machine Learning · KI-Agenten · Schulungen

Erhard Rainer · Schönaugasse 106 · 8020 Graz · info@think-bigger.at

Inhalt

Inhaltsverzeichnis 2
1 - Geschichte und Versionsgeschichte 3
2 - Technische Architektur 4
3 - X++ Programmiersprache und Entwicklungsumgebung 6
4 - Datenbankarchitektur auf SQL-Server-Ebene 9
5 - Module im Detail 12
6 - Security-Modell und Benutzerrechte 20
7 - Integration: AIF, Services und Schnittstellen 21
8 - Reporting: SSRS, Management Reporter und OLAP 23
9 - Performance und Datenbankoptimierung 24
10 - Migration nach Dynamics 365 Finance & Operations 25
11 - AX 2012 als Datenquelle fur moderne Analytics 26
12 - Fazit 30
13 - Literatur und Ressourcen 31

> **Kurzfassung:** Microsoft Dynamics AX 2012 ist eines der komplexesten und leistungsfähigsten ERP-Systeme, die je fur den Mittelstand und Konzerne entwickelt wurden. Wer AX 2012 produktiv betreibt, entwickelt oder als Datenquelle fur moderne Cloud-Architekturen nutzt, braucht ein tiefes Verstandnis seiner Architektur, seines Datenbankschemas und seiner Integrationsmoglichkeiten. Dieser Artikel liefert dieses Verstandnis — von der technischen Schichtarchitektur uber die wichtigsten Tabellen bis zum Migrationspfad nach Dynamics 365 Finance & Operations.

Microsoft Dynamics AX 2012 — kurz AX 2012 — ist die letzte große On-Premises-Generation der Axapta/Dynamics-AX-Produktlinie. Microsoft hat zwischen 2011 und 2016 drei Release-Versionen veroffentlicht (R1, R2, R3). Heute laufen weltweit noch tausende von Produktionsinstanzen — oft als System of Record, aus dem Daten in moderne Cloud-Plattformen wie Microsoft Fabric, Azure Synapse oder Power BI migriert werden.

Im Artikelverbund ist dieser Text der technische AX-Deep-Dive. Der Microsoft-ERP-Einstieg ordnet Dynamics AX, NAV, GP, SL, Business Central und Dynamics 365 als Produktfamilie ein. Der ERP-Hub behandelt die Anbieter- und Auswahlperspektive; der SAP-Hub bleibt ein Nachbar für SAP-spezifische Architekturen, nicht die Heimat dieses Microsoft-ERP-Artikels.

Dieser Artikel betrachtet AX 2012 aus mehreren Perspektiven:

- **technische Architektur** (AOS, AOT, Schichten, Metadaten)
- **X++ als Programmiersprache** und Entwicklungsumgebung
- **Datenbankschema** — SQL-Server-Ebene, RecId, DataAreaId, Partitionierung
- **Module im Detail** — mit den wichtigsten Tabellen je Bereich
- **Security-Modell** — Rollen, Berechtigungen, Datenebene
- **Integration** — AIF, Services, Schnittstellen
- **Reporting** — SSRS, Management Reporter, OLAP-Cubes
- **Migration nach Dynamics 365 FO** — Upgrade-Pfade, Datenmigration
- **AX 2012 als Datenquelle** fur Fabric und moderne Analytics-Plattformen

---

Inhaltsverzeichnis

- 1 - Geschichte und Versionsgeschichte
- 2 - Technische Architektur
- 3 - X++ Programmiersprache und Entwicklungsumgebung
- 4 - Datenbankarchitektur auf SQL-Server-Ebene
- 5 - Module im Detail
- 6 - Security-Modell und Benutzerrechte
- 7 - Integration: AIF, Services und Schnittstellen
- 8 - Reporting: SSRS, Management Reporter und OLAP
- 9 - Performance und Datenbankoptimierung
- 10 - Migration nach Dynamics 365 Finance & Operations
- 11 - AX 2012 als Datenquelle fur moderne Analytics
- 12 - Fazit
- 13 - Literatur und Ressourcen

---

1 - Geschichte und Versionsgeschichte

1.1. Von Axapta zu Dynamics AX

Die Geschichte von Microsoft Dynamics AX ist eine Geschichte von Unternehmensakquisitionen und konsequenter Produktweiterentwicklung.

1983 — Damgaard Data: Die danische Firma Damgaard Data wurde von den Brudern Erik und Preben Damgaard gegrundet. Ihr Ziel war eine Business-Software-Plattform, die europaische Buchungsstandards nativ unterstutz.

1998 — Axapta 1.0: Mit Axapta erschuf Damgaard Data ein vollstandig objektorientiertes ERP-System mit einer eigens entwickelten Programmiersprache (X++), einem eigenem Application Server und einem eigenem Metadaten-Repository (AOT). Das war fur die damalige Zeit aussergewohnlich.

2000 — Fusion mit Navision: Damgaard und Navision fusionierten zu Navision Software. Axapta und Navision liefen als getrennte Produkte weiter.

2002 — Microsoft-Akquisition: Microsoft erwarb Navision Software fur 1,45 Milliarden US-Dollar und integrierte die Produkte in die neue Microsoft Business Solutions (MBS) Sparte, spater umbenannt in Microsoft Dynamics.

2003 — Microsoft Axapta 3.0: Erste Version unter Microsoft-Dach, noch unter dem Axapta-Namen.

2006 — Microsoft Dynamics AX 4.0: Erste Version mit vollstandigem Dynamics-Branding. Einfuhrung des .NET Business Connector.

2008 — Microsoft Dynamics AX 2009: Einfuhrung von Enterprise Portal (webbasiertes Frontend uber SharePoint) und Windows Communication Foundation (WCF) basierte Services.

2011 — Microsoft Dynamics AX 2012: Die wichtigste und umfangreichste Version der On-Premises-Linie. Zahlreiche Architekturanderungen: Multi-Company per Partition, neue Sicherheitsarchitektur mit Rollen und Berechtigungen, nativer SSRS-Reporting-Stack, verbesserter Application Object Tree.

2013 — AX 2012 R2: Einfuhrung von Data Partitioning (mehrere logische Instanzen in einer AX-Datenbank), verbessertes Upgrade-Framework, Erweiterung des Retail-Moduls.

2015 — AX 2012 R3: Letzte Release-Aktualisierung der 2012-Linie. Verbesserungen in Lagerverwaltung (Advanced Warehouse Management), Transport Management, Retail und Cloud-Integration via Azure. Ende des Mainstream Supports fur AX 2012 am 10. Oktober 2021, Extended Support bis Oktober 2023.

2016 — Dynamics 365 Finance & Operations (D365 FO): Microsoft veroffentlichte die Cloud-native Nachfolge-Plattform auf Azure-Basis. D365 FO ist die direkte Weiterentwicklung von AX 2012, aber vollstandig cloud-nativ, mit anderem Deployment-Modell und anderem Entwicklungsparadigma. Eine breitere Einordnung der modernen App-Familie steht im Deep-Dive zu Dynamics 365.

1.2. Versionsübersicht

Version	Jahr	Wichtigste Neuerungen
Axapta 1.0	1998	Erstveröffentlichung, X++, AOT, AOS
Axapta 3.0	2002	Unter Microsoft, .NET-Integration
Dynamics AX 4.0	2006	Microsoft-Branding, .NET BC
Dynamics AX 2009	2008	Enterprise Portal, WCF Services

Version	Jahr	Wichtigste Neuerungen
Dynamics AX 2012	2011	Neue Security, SSRS, nativer Stack
Dynamics AX 2012 R2	2013	Data Partitioning, erweiterter Retail
Dynamics AX 2012 R3	2015	Warehouse Management, letzte On-Prem-Version
Dynamics 365 FO	2016	Cloud-native Nachfolge

1.3. Support-Status und Relevantzim Heute

Obwohl AX 2012 offiziell aus dem Extended Support heraus ist, laufen weltweit noch tausende Produktionsinstanzen. Die Hauptgrunde:

- **Migration ist teuer und riskant**: Eine D365-FO-Migration ist ein Grossprojekt
- **AX 2012 funktioniert**: Viele Unternehmen haben keine unmittelbaren Probleme
- **On-Premises-Anforderungen**: Manche Branchen oder Regulierungen bevorzugen On-Premises-Deployment
- **Anpassungen**: viele Unternehmen haben aufwendige ISV-Losungen oder eigene Anpassungen, die migriert werden mussen

Fur IT-Architekten und Datenentwickler ist AX 2012 heute besonders relevant als **Datenquelle** fur moderne Cloud-Analytics-Plattformen — als Basis fur die Extraktion nach Microsoft Fabric, Azure Synapse oder Power BI.

---

2 - Technische Architektur

2.1. Dreiteilige Client-Server-Architektur

AX 2012 verwendet eine **dreiteilige Architektur**:

```
[Client]  ↔  [AOS - Application Object Server]  ↔  [SQL Server Database]
```

- **Client**: Windows-Client (Rich Client) oder Enterprise Portal (Web)
- **AOS**: zentraler Anwendungsserver, fuhrt Geschaftslogik aus
- **SQL Server**: relationale Datenbank, speichert alle Daten

Der AOS ist das Herzstuck. Er ist:

- der einzige Zugriffspunkt fur die Datenbank (Clients greifen nie direkt auf SQL zu)
- Ausgabeort fur alle Berechnungen und Geschaftsregeln
- Verwaltungsort fur Sessions, Sicherheit und Cache
- Host fur AIF (Application Integration Framework) Services

2.2. Application Object Tree (AOT)

Der **Application Object Tree** ist eines der herausragendsten Konzepte von AX. Er ist ein **zentrales Repository fur alle Anwendungsobjekte**:

- **Tabellen** (physische Tabellen und Views)
- **Klassen** (Geschaftslogik in X++)
- **Formulare** (Windows-UI-Masken)

- **Reports** (SSRS-Reports)
- **Menus** und Menu Items
- **Data Dictionary** (Datentypen, Extended Data Types, Enums)
- **Projekte** (Entwickler-Arbeitsbereiche)
- **Konfigurationsschlussel** (Feature-Toggles)
- **Sicherheitsobjekte** (Berechtigungen, Rollen, Policies)

Der AOT ist nicht nur ein Editor, sondern das **Single Point of Truth** fur alles, was in AX existiert. Tabellen, die im AOT angelegt werden, werden automatisch in der SQL-Datenbank erstellt. Anderungen am Schema im AOT synchronisieren die SQL-Datenbank.

2.3. Schichtarchitektur

AX 2012 verwendet ein **Schichtmodell** fur Anpassungen. Es gibt 8 Schichten (von niedrigster zu hochster Prioritat):

Schicht	Kürzel	Zweck
SYS	SYS	Microsoft Standardcode (unveranderlich)
SYP	SYP	Microsoft Patches
GLS	GLS	Globale Lokalisierungen (Microsoft)
GLO	GLO	Globale Lokalisierungen (Partner)
LOS	LOS	Lokale Lokalisierungen (Microsoft)
LOC	LOC	Lokale Lokalisierungen (Partner)
USR	USR	Kundenspezifische Anpassungen
VAR	VAR	ISV-/Partner-Anpassungen
CUS	CUS	Kundenspezifische Uberschreibungen
BUS	BUS	Business-Partner-Erweiterungen

Wie Schichten funktionieren:

Wenn ein Objekt (z.B. eine Klasse) in SYS definiert ist und in CUS uberschrieben wird, sieht AX beim Aufruf nur die CUS-Version. SYS-Code bleibt unverandert. Das ermoglicht Upgrades ohne Verlust von Anpassungen — theoretisch. In der Praxis ist das Upgrade-Management trotzdem aufwendig.

Praktische Implikation: Bei SQL-Abfragen spielen Schichten keine Rolle — Sie sehen nur die Tabellendaten, nicht Metadaten-Schichten. Schichten sind AOS-intern.

2.4. Konfigurationsschlussel

Konfigurationsschlussel (Configuration Keys) sind Feature-Toggles in AX. Sie steuern, welche Module und Funktionen aktiviert sind:

- jede Tabelle, jedes Feld, jede Maske hat einen oder mehrere Konfigurationsschlussel
- wird ein Konfigurationsschlussel deaktiviert, verschwinden die zugehorigen Felder und Masken
- **Lizenzcode**: Konfigurationsschlussel sind an Lizenzkeys gebunden

Fur die Datenmigration wichtig: **Felder, die durch deaktivierte Konfigurationsschlussel ausgeblendet sind, existieren trotzdem in der SQL-Datenbank** — sie sind nur in der AX-Oberflache nicht sichtbar.

2.5. Data Dictionary

Das **Data Dictionary** im AOT definiert:

- **Extended Data Types (EDT)**: Ableitung von Basisdatentypen mit zusatzlicher Semantik (z.B. `CustAccount` als EDT auf Basis von `AccountNum` als String)
- **Base Enumerations (Base Enums)**: numerische Aufzahlungen mit Bezeichnungen
- **Table Collections**: logische Gruppierungen von Tabellen

EDTs sind fur Entwickler wichtig, weil sie die fachliche Bedeutung eines Feldes ohne SQL-Abfrage der Feldnamen erschlossen werden kann.

---

3 - X++ Programmiersprache und Entwicklungsumgebung

3.1. X++ — Eine eigene Sprache

X++ (gesprochen: "X Plus Plus") ist die Programmiersprache von Dynamics AX. Sie wurde von Damgaard Data fur Axapta entwickelt und ist seither das Kernentwicklungswerkzeug.

X++ ist:

- **objektorientiert** (Klassen, Vererbung, Interfaces, Polymorphismus)
- **SQL-integriert** (SELECT, INSERT, UPDATE, DELETE als native Sprachkonstrukte)
- **stark typisiert**
- **garbage-collected** (keine manuell Speicherverwaltung)
- **interpretiert auf dem AOS** (nicht zu verwechseln mit kompiliertem .NET-Code)

3.2. Syntax-Grundlagen

```
// Klassendefinition
class MyBusinessLogic extends SysRunBase
{
    CustAccount     custAccount;
    AmountMST       threshold;
    // Statische Methode
    static MyBusinessLogic construct()
    {
        return new MyBusinessLogic();
    }
    // Hauptmethode
    public void run()
    {
        CustTable   custTable;
        // SELECT mit WHERE-Bedingung
        select custTable
            where custTable.AccountNum == custAccount;
        if (custTable)
        {
            info(strFmt("Kunde gefunden: %1 - %2",
                custTable.AccountNum,
                custTable.name()));
        }
        else
        {
            warning("Kein Kunde gefunden.");
        }
```

```
    }
}
```

3.3. SQL in X++

X++ integriert SQL-Operationen direkt als Sprachkonstrukte:

```
// Einfacher SELECT
CustTable   custTable;
select firstOnly custTable
    where custTable.AccountNum == '10001';

// SELECT mit ORDER BY und Range
InventTrans inventTrans;
while select inventTrans
    order by ItemId, TransDate
    where inventTrans.ItemId == 'A0001'
       && inventTrans.TransDate >= mkDate(1, 1, 2026)
{
    info(strFmt("Item: %1, Menge: %2", inventTrans.ItemId, inventTrans.Qty));
}

// JOIN zwischen Tabellen
CustTable   custTable;
CustTrans   custTrans;
while select custTable
    join custTrans
        where custTrans.AccountNum == custTable.AccountNum
           && custTrans.TransDate >= mkDate(1, 1, 2026)
{
    // Verarbeitung
}

// INSERT
InventTable inventTable;
inventTable.ItemId      = 'NEWITEM';
inventTable.ItemName    = 'Neuer Artikel';
inventTable.ItemType    = ItemType::Item;
inventTable.insert();

// UPDATE
update_recordset inventTable
    setting ItemName = 'Geaenderter Name'
    where inventTable.ItemId == 'NEWITEM';

// DELETE
delete_recordset inventTable
    where inventTable.ItemId == 'NEWITEM';
```

Wichtige X++-SQL-Besonderheiten:

- `select firstOnly` — liest nur den ersten Datensatz (effizienter als ohne)
- `forupdate` — setzt Datensatzsperre fur spatere Anderungen
- `crossCompany` — liest uber mehrere Companies (DataAreas)
- Kein nativer SQL MERGE in X++ — Upserts werden prozedural mit SELECT + INSERT/UPDATE implementiert

3.4. MorphX Entwicklungsumgebung

MorphX ist die integrierte Entwicklungsumgebung (IDE) von AX, direkt in den Windows-Client eingebaut.

MorphX bietet:

- **AOT-Editor**: Bearbeitung aller Anwendungsobjekte
- **X++ Code-Editor**: mit Syntax-Highlighting und IntelliSense
- **Debugger**: Breakpoints, Variable-Watch, Call Stack
- **Label-Editor**: Verwaltung mehrsprachiger Texte
- **Cross-Reference**: Abhangigkeitsanalyse zwischen Objekten
- **Best Practice Checker**: statische Code-Analyse

3.5. Wichtige Klassen und Konzepte

RunBase und SysRunBase

Viele AX-Batches und Jobs erben von `SysRunBase`:

```
class MyBatchJob extends SysRunBase
{
    // Pflichtmethode: Ausgabe-Beschreibung
    public ClassDescription caption()
    {
        return "Mein Batch-Job";
    }
    // Pflichtmethode: Hauptlogik
    public void run()
    {
        // Geschaftslogik
    }
    // Factory-Methode
    static void main(Args _args)
    {
        MyBatchJob job = MyBatchJob::construct();
        if (job.prompt())
        {
            job.run();
        }
    }
}
```

RecordInsertList (Bulk Insert)

```
RecordInsertList    ril = new RecordInsertList(tableNum(InventTrans));
InventTrans         inventTrans;

for (i = 0; i < 1000; i++)
{
    inventTrans.clear();
    inventTrans.ItemId  = 'A0001';
    inventTrans.Qty     = i;
    ril.add(inventTrans);
}
ril.insertDatabase();
```

`RecordInsertList` schreibt Datensatze in Chunks in die Datenbank — deutlich performanter als 1000 einzelne `insert()`-Aufrufe.

SysQuery und QueryRun

```
Query           query;
QueryBuildDataSource qbds;
QueryBuildRange     qbr;
QueryRun        queryRun;
CustTable       custTable;
```

```
query   = new Query();
qbds    = query.addDataSource(tableNum(CustTable));
qbr     = qbds.addRange(fieldNum(CustTable, AccountNum));
qbr.value(SysQuery::range('10000', '19999'));

queryRun = new QueryRun(query);
while (queryRun.next())
{
    custTable = queryRun.get(tableNum(CustTable));
    // Verarbeitung
}
```

---

4 - Datenbankarchitektur auf SQL-Server-Ebene

4.1. SQL Server als Datenbankschicht

AX 2012 verwendet **Microsoft SQL Server** (2008 R2, 2012, 2014, 2016 je nach AX-Version und Patch) als relationale Datenbank. Die AX-Datenbank ist eine gewohnliche SQL-Server-Datenbank, auf die man mit SSMS, TSQL, oder modernen ETL-Tools direkt zugreifen kann.

Wichtige Eigenheiten:

- alle AX-Tabellen haben bestimmte **technische Pflichtfelder**
- das Schema ist sehr flach (keine ausgepragte Normalisierung uber viele Ebenen)
- viele Tabellen haben keine expliziten SQL-Fremdschlusselconstraints — Referenzintegritat wird auf AOS-Ebene erzwungen
- Indexe sind im AOT definiert und werden in SQL synchronisiert

4.2. Technische Pflichtfelder in jeder AX-Tabelle

Jede Tabelle in AX 2012 hat systemseitig erzeugte technische Felder:

Feldname	SQL-Typ	Bedeutung
`RecId`	bigint	eindeutiger Primarschlussel pro Tabelle, automatisch vergeben
`DataAreaId`	nvarchar(4)	Unternehmenskennung (Company), Mehrfirmen-Kontext
`Partition`	bigint	Daten-Partition (nur ab AX 2012 R2)
`RecVersion`	int	optimistisches Locking (wird bei jedem Update hochgezahlt)
`CreatedBy`	nvarchar(8)	AX-Benutzer, der den Datensatz erstellt hat
`CreatedDateTime`	datetime	Zeitpunkt der Erstellung
`ModifiedBy`	nvarchar(8)	AX-Benutzer, der zuletzt geandert hat
`ModifiedDateTime`	datetime	Zeitpunkt der letzten Anderung

RecId:

- `RecId` ist der einzige garantiert eindeutige technische Schlussel pro Tabelle
- er wird von einem zentralen **Sequence-Generator** vergeben
- er ist nicht monoton uber Tabellen hinweg — zwei Tabellen konnen denselben RecId-Wert haben
- er ist aber pro Tabelle eindeutig und unveranderlich (nach Insert)
- RecId 0 bedeutet "kein Datensatz" und wird als NULL-Aquivalent fur Referenzfelder genutzt

DataAreaId:

- steuert, welche Firma (Company) ein Datensatz gehort
- die meisten Geschaftsdatentabellen haben `DataAreaId`
- systemweite Tabellen (wie `UserInfo`, `SysConfig`) haben kein `DataAreaId`
- beim direkten SQL-Zugriff muss `DataAreaId` immer als Filter mitgefuhrt werden

Partition (ab AX 2012 R2):

- erlaubt mehrere logische AX-Instanzen in einer physischen Datenbank
- jede Partition hat eine eigene Konfiguration, Benutzer und Firmen
- beim direkten SQL-Zugriff muss `Partition` als Filter mitgefuhrt werden, wenn mehrere Partitionen existieren
- `SELECT * FROM CUSTTABLE WHERE DATAAREAID = 'USMF' AND PARTITION = 5637144576`

4.3. Namenskonventionen

AX-Tabellennamen folgen bestimmten Mustern:

Prafix	Bereich	Beispiel
Cust	Debitoren	CustTable, CustTrans, CustInvoiceJour
Vend	Kreditoren	VendTable, VendTrans, VendInvoiceJour
Invent	Lagerverwaltung	InventTable, InventTrans, InventDim
Sales	Vertrieb	SalesTable, SalesLine, SalesParmTable
Purch	Einkauf	PurchTable, PurchLine, PurchParmTable
Proj	Projekt	ProjTable, ProjTrans, ProjEmplTrans
HRM	Human Resources	HRMWorker, HRMEmployment
EcoRes	Produktstamm	EcoResProduct, EcoResProductDimensionGroup
WMS	Warehouse Mgmt	WMSLocation, WMSPickingRoute
Ledger	Hauptbuch	LedgerJournalTrans, LedgerTrans
Tax	Steuern	TaxTable, TaxTrans
Sys	System	SysConfig, SysSetupLog

4.4. Datenbankschema — Kerntabellen

System- und Konfigurationstabellen

Tabelle	Inhalt
`SysConfig`	Systemkonfiguration, AX-Einstellungen
`SysUserInfo`	Benutzerprofil-Erweiterungen
`UserInfo`	AX-Benutzer (Login, Sprache, Zeitzone)
`DataArea`	Firmenstamm (alle Companies/DataAreaIds)
`PartitionSys`	Partitions-Konfiguration (ab R2)
`SysBatchTable`	Batch-Job-Definitionen
`SysBatchJob`	Batch-Server-Konfiguration

Tabelle	Inhalt
`SysLabel`	Label-Texte (Mehrsprachigkeit)
`SysLabelFile`	Label-Dateien

Tabellen fur RecId-Tracking

Tabelle	Inhalt
`SystemSequences`	Sequenz-Generator fur RecIds (enthalt nächste freie RecId)

4.5. Direkte SQL-Abfragen auf AX-Datenbank

Wichtigste Regel: **Immer DataAreaId und Partition filtern.**

```
-- Kundenstamm einer bestimmten Company
SELECT
    RecId,
    AccountNum,
    Name,
    CustGroup,
    Currency,
    PaymTermId,
    CreditMax,
    ModifiedDateTime
FROM CustTable
WHERE DataAreaId = 'USMF'
  AND Partition  = 5637144576  -- nur wenn mehrere Partitionen existieren
ORDER BY AccountNum;

-- Offene Debitorenposten mit Verknüpfung zum Kundenstamm
SELECT
    ct.AccountNum,
    ct.Name,
    ctr.Voucher,
    ctr.TransDate,
    ctr.AmountMST,
    ctr.AmountCur,
    ctr.CurrencyCode,
    ctr.Invoice,
    ctr.Due,
    ctr.Closed
FROM CustTrans ctr
INNER JOIN CustTable ct
    ON ct.AccountNum = ctr.AccountNum
   AND ct.DataAreaId = ctr.DataAreaId
   AND ct.Partition  = ctr.Partition
WHERE ctr.DataAreaId = 'USMF'
  AND ctr.Open_      = 1  -- Open_ wegen reserviertem SQL-Keyword "Open"
ORDER BY ctr.AccountNum, ctr.TransDate;
```

Haufige Fallstricke:

- Feldnamen mit reservierten SQL-Schlusselwortern werden in AX mit Unterstrich suffixiert: `Open_`, `Level_`, `Group_`
- `nvarchar`-Felder in AX sind oft viel breiter als der tatsachliche Inhalt (z.B. `nvarchar(20)` fur eine 4-stellige Nummer)
- DateTime-Felder in AX sind UTC ohne Zeitzoneninfo — Anzeige im Client passt sich an Benutzer-Zeitzone an

---

5 - Module im Detail

5.1. Finanzmanagement

Das Finanzmanagement-Modul ist das Herzstuck von AX 2012. Es deckt alle buchhalterischen Prozesse ab und ist das meistgenutzte Modul in fast allen AX-Installationen.

5.1.1. Hauptbuch (General Ledger)

Wichtigste Konzepte:

- **Kontenplan (Chart of Accounts)**: wird pro Company definiert, kann aber geteilt werden
- **Finanzdimensionen (Financial Dimensions)**: erweiterbare Kostenstellenstruktur (z.B. Kostenstelle, Profit Center, Projekt)
- **Finanzdimensionssätze (Dimension Sets)**: Kombinationen von Dimensionen fur Berichtszwecke
- **Buchungsschicht (Ledger Layer)**: jede Transaktion kann in mehreren Buchungsebenen existieren (Current, Tax, Operations, ...)
- **Geschlossene Perioden (Closed Periods)**: verhindert Buchungen in vergangene Perioden

Wichtigste Tabellen Hauptbuch:

Tabelle	Inhalt
`MainAccount`	Sachkonten-Stamm (Kontenplan)
`MainAccountCategory`	Konten-Kategorien fur Berichte
`LedgerJournalTable`	Journal-Kopfdaten (Buchungsjournal)
`LedgerJournalTrans`	Journal-Buchungszeilen
`GeneralJournalEntry`	generierte Buchungssätze (nach dem Buchen)
`GeneralJournalAccountEntry`	einzelne Kontozeilenbuchungen
`LedgerTrans`	gebuchte Hauptbuch-Transaktionen (historisch)
`LedgerTransSettlement`	Ausgleichsbuchungen zwischen LedgerTrans-Einträgen
`DimensionAttributeValueCombination`	Finanzdimensions-Kombinationen
`DimensionAttributeValueSet`	Dimensionswerte-Sets
`LedgerPeriod`	Buchungsperioden
`FiscalCalendar`	Geschäftsjahres-Kalender

Beispiel-Abfrage: Alle Hauptbuchbuchungen eines Kontos

```
SELECT
    gle.JournalNumber,
    gle.AccountingDate,
    gaae.MainAccountId,
    gaae.AccountingCurrencyAmount,
    gaae.TransactionCurrencyAmount,
    gaae.Text,
    gaae.PostingType
FROM GeneralJournalEntry gle
INNER JOIN GeneralJournalAccountEntry gaae
    ON gaae.GeneralJournalEntry = gle.RecId
    AND gaae.Partition = gle.Partition
WHERE gle.DataAreaId       = 'USMF'
  AND gle.Partition        = 5637144576
```

```
  AND gaae.MainAccountId   = '110110'
  AND gle.AccountingDate  >= '2026-01-01'
ORDER BY gle.AccountingDate;
```

5.1.2. Debitorenbuchhaltung (Accounts Receivable)

Wichtigste Tabellen:

Tabelle	Inhalt
`CustTable`	Kundenstamm
`CustGroup`	Kundengruppen
`CustTrans`	Debitorentransaktionen (Buchungsposten)
`CustTransOpen`	offene Debitorenposten
`CustInvoiceJour`	Debitorenrechnungsjournal (Kopfdaten)
`CustInvoiceTrans`	Debitorenrechnungszeilen
`CustPaymJour`	Zahlungsjournal (Kopf)
`CustPaymLine`	Zahlungszeilen
`CustSettlement`	Ausgleichsbuchungen
`CustCollectionLetterJour`	Mahnungskopf
`CustInvoiceTable`	Kundenauftrags-Rechnungskopf
`CustLedgerAccounts`	Kontierzungseinstellungen Debitoren

Wichtige Felder in CustTable:

Feld	Typ	Bedeutung
`AccountNum`	nvarchar(20)	Kundennummer (Geschäftsschlüssel)
`Name`	nvarchar(60)	Kundenname
`CustGroup`	nvarchar(10)	Kundengruppe
`Currency`	nvarchar(3)	Standardwährung
`PaymTermId`	nvarchar(10)	Zahlungsbedingungen
`CreditMax`	numeric(32,6)	Kreditlimit
`BlockedC`	int	gesperrt (Enum: Nein/Rechnungen/Alles)
`OneTimeCustomer`	int	Einmalkunde
`DlvMode`	nvarchar(10)	Lieferart
`LogisticsPostalAddress`	bigint	Adress-RecId (Verweis auf LogisticsPostalAddress)
`ModifiedDateTime`	datetime	letzte Änderung (UTC)

5.1.3. Kreditorenbuchhaltung (Accounts Payable)

Wichtigste Tabellen:

Tabelle	Inhalt
`VendTable`	Lieferantenstamm
`VendGroup`	Lieferantengruppen

Tabelle	Inhalt
`VendTrans`	Kreditorentransaktionen
`VendTransOpen`	offene Kreditorenposten
`VendInvoiceJour`	Lieferantenrechnungsjournal (Kopf)
`VendInvoiceTrans`	Lieferantenrechnungszeilen
`VendPaymJour`	Zahlungsjournal (Lieferanten)
`VendSettlement`	Lieferanten-Ausgleichsbuchungen
`PurchTable`	Bestellkopf
`PurchLine`	Bestellzeilen

5.1.4. Anlagenbuchhaltung (Fixed Assets)

Wichtigste Tabellen:

Tabelle	Inhalt
`AssetTable`	Anlagenstamm
`AssetGroup`	Anlagengruppen
`AssetTrans`	Anlagenbuchungen
`AssetBook`	Abschreibungsbücher pro Anlage
`AssetBookTable`	Abschreibungsbuch-Definitionen
`AssetDepRecordTrans`	Abschreibungserfassungen

5.2. Lieferkettenmanagement

5.2.1. Artikel- und Produktstamm

Hinweis zur AX 2012 Produktarchitektur:

AX 2012 fuhrt eine zweistufige Produktstruktur ein:

- **EcoResProduct**: globaler Produktstamm (company-ubergreifend)
- **InventTable**: company-spezifische Artikelstammdaten

Das ist ein wesentlicher Unterschied zu fruheren AX-Versionen, wo `InventTable` der einzige Produktstamm war.

Wichtigste Produktstamm-Tabellen:

Tabelle	Inhalt
`EcoResProduct`	globaler Produktstamm (company-unabhangig)
`EcoResDistinctProduct`	freigegebene Produkte (Produktvarianten-Konzept)
`EcoResProductTranslation`	mehrsprachige Produktbeschreibungen
`EcoResCategory`	Produktkategorien
`EcoResCategoryHierarchy`	Kategoriehierarchien
`EcoResAttribute`	Produktattribute
`EcoResAttributeValue`	Attributwerte

Tabelle	Inhalt
`EcoResProductDimensionGroup`	Produktdimensionsgruppen (Farbe, Größe, Stil)
`InventTable`	freigegebene Produkte (company-spezifisch)
`InventTableModule`	Modulparameter pro Artikel (Einkauf, Verkauf, Lager)
`InventItemGroup`	Artikelgruppen (fur Kontierung)
`InventModelGroup`	Lagermodellgruppen (FIFO, LIFO, Standardpreis)
`InventDimGroup`	Lagerdimensionsgruppen

5.2.2. Lagerverwaltung

Lagerdimensionen in AX 2012:

AX 2012 verwendet ein ausgeklugeltes **Dimensionsmodell** fur Lagerartikel. Jeder Lagerposten wird durch eine Kombination aus Lagerdimensionen beschrieben:

Produkt-Dimensionen:

- `ConfigId` — Konfiguration
- `InventColorId` — Farbe
- `InventSizeId` — Größe
- `InventStyleId` — Stil

Lager-Dimensionen:

- `InventSiteId` — Standort
- `InventLocationId` — Lager
- `InventBatchId` — Charge
- `WmsLocationId` — Lagerplatz
- `WmsPalletId` — Palette
- `InventSerialId` — Seriennummer

Nachverfolgungs-Dimensionen:

- `LicensePlateId` — Ladungsträger (ab R3)

Alle Dimensionen werden in `InventDim` kombiniert gespeichert. Jede eindeutige Kombination hat eine eigene RecId in `InventDim`.

Wichtigste Lagertabellen:

Tabelle	Inhalt
`InventDim`	Lagerdimensionskombinationen
`InventTrans`	alle Lagertransaktionen (Zugänge, Abgänge)
`InventTransOrigin`	Ursprung einer Lagertransaktion (z.B. Bestellzeile)
`InventSum`	aktueller Lagerbestand (aggregiert)
`InventOnhand`	Bestandsübersicht
`InventJournalTable`	Lagerjournalkopf
`InventJournalTrans`	Lagerjournalzeilen

Tabelle	Inhalt
`InventSite`	Standorte
`InventLocation`	Lager
`WMSLocation`	Lagerplatze
`WMSPallet`	Paletten

Wichtige Felder in InventTrans:

Feld	Bedeutung
`ItemId`	Artikelnummer
`InventDimId`	Verweis auf InventDim (Dimensionskombination)
`Qty`	Menge
`StatusIssue`	Status des Abgangs (Sold, Deducted, OnOrder, ...)
`StatusReceipt`	Status des Zugangs (Received, Purchased, ...)
`CostAmountPosted`	gebuchter Wertansatz
`DatePhysical`	physisches Buchungsdatum
`DateFinancial`	finanzielles Buchungsdatum
`TransType`	Transaktionstyp (Bestellung, Auftrag, Journal, ...)
`InventTransOrigin`	Verweis auf Ursprungsdokument

5.2.3. Verkauf (Sales)

Auftragsfluss in AX:

```
Angebot (SalesQuotationTable/Line)
→ Auftrag (SalesTable/Line)
→ Lieferschein (CustPackingSlipJour/Trans)
→ Rechnung (CustInvoiceJour/Trans)
```

Wichtigste Verkaufs-Tabellen:

Tabelle	Inhalt
`SalesTable`	Verkaufsauftragskopf
`SalesLine`	Verkaufsauftragspositionen
`SalesParmTable`	Parameter fur Buchungsvorgänge
`SalesParmLine`	Buchungsparameter-Zeilen
`CustPackingSlipJour`	Lieferscheinkopf
`CustPackingSlipTrans`	Lieferscheinpositionen
`SalesQuotationTable`	Angebotskopf
`SalesQuotationLine`	Angebotspositionen
`SalesAgreementHeader`	Kaufverträge (Kopf)
`SalesAgreementLine`	Kaufverträge (Zeilen)
`ReturnTable`	Rücksendungsauftragskopf

Wichtige Felder in SalesTable:

Feld	Bedeutung
`SalesId`	Auftrags-ID (Geschäftsschlüssel)
`CustAccount`	Kundennummer
`SalesStatus`	Status (Offen, Geliefert, Fakturiert, Storniert)
`SalesType`	Auftragstyp (Auftrag, Retoure, Projekt)
`DeliveryDate`	gewunschtes Lieferdatum
`ShippingDateRequested`	angefordertes Versanddatum
`CurrencyCode`	Währung
`SalesBalance`	Auftragssumme
`InvoiceAccount`	Rechnungskunde (wenn abweichend)
`ModifiedDateTime`	letzte Änderung

5.2.4. Einkauf (Procurement)

Bestellungsfluss in AX:

```
Bestellanforderung (PurchReqTable/Line)
→ Anfrage/Ausschreibung (PurchRFQTable/Line)
→ Bestellung (PurchTable/Line)
→ Wareneingang (VendPackingSlipJour/Trans)
→ Lieferantenrechnung (VendInvoiceJour/Trans)
```

Wichtigste Einkaufs-Tabellen:

Tabelle	Inhalt
`PurchTable`	Bestellkopf
`PurchLine`	Bestellpositionen
`VendPackingSlipJour`	Lieferschein Lieferant (Kopf)
`VendPackingSlipTrans`	Lieferschein Lieferant (Zeilen)
`PurchReqTable`	Bestellanforderung (Kopf)
`PurchReqLine`	Bestellanforderung (Zeilen)
`PurchAgreementHeader`	Rahmenverträge Einkauf (Kopf)
`PurchAgreementLine`	Rahmenverträge Einkauf (Zeilen)

Wichtige Felder in PurchLine:

Feld	Bedeutung
`PurchId`	Bestellnummer
`LineNum`	Positionsnummer
`ItemId`	Artikelnummer
`PurchQty`	bestellte Menge
`RemainPurchFinancial`	verbleibende Menge (finanziell)

Feld	Bedeutung
`PurchPrice`	Einkaufspreis
`PurchUnit`	Einkaufseinheit
`DeliveryDate`	Liefertermin
`LineAmount`	Positionsbetrag
`InventDimId`	Lagerdimensionen
`ModifiedDateTime`	letzte Änderung

5.3. Fertigung (Production)

AX 2012 bietet drei Fertigungsarten:

- **Diskrete Fertigung** (Production Orders)
- **Lean Manufacturing** (Kanban)
- **Prozessfertigung** (Formula-basiert, nur R2/R3)

Wichtigste Fertigungs-Tabellen:

Tabelle	Inhalt
`ProdTable`	Produktionsauftragskopf
`ProdBOM`	Stücklisten-Positionen des Produktionsauftrags
`ProdRoute`	Arbeitsplan-Vorgänge des Auftrags
`BOMTable`	Stücklistenkopf (Stamm)
`BOMVersion`	Stücklisten-Versionen
`BOM`	Stücklistenpositionen (Stamm)
`RouteTable`	Arbeitspläne (Stamm)
`RouteVersion`	Arbeitsplan-Versionen
`RouteOpr`	Arbeitsplanvorgänge
`WrkCtrTable`	Arbeitsplatze und Ressourcengruppen
`ProdJournalBOM`	Stücklisten-Entnahmejournal
`ProdJournalRoute`	Rückmeldungsjournal
`PmfCobyItem`	Kuppelprodukte (Prozessfertigung)

Wichtige Felder in ProdTable:

Feld	Bedeutung
`ProdId`	Produktionsauftragsnummer
`ItemId`	Fertigartikel
`ProdQty`	Auftragsmenge
`RemainInventPhysical`	verbleibende physische Menge
`SchedStatus`	Planungsstatus
`ProdStatus`	Auftragsstatus (Erstellt, Geschätzt, Freigegeben, Gemeldet, Abgeschlossen)

Feld	Bedeutung
`DlvDate`	Liefertermin
`StartedDate`	tatsächliches Startdatum
`InventDimId`	Zieldimensionen

5.4. Projektmanagement und Buchhaltung

Wichtigste Projekt-Tabellen:

Tabelle	Inhalt
`ProjTable`	Projektstamm
`ProjGroup`	Projektgruppen
`ProjCategory`	Projektkategorien (Stunden, Ausgaben, Artikel)
`ProjTrans`	Projekttransaktionen
`ProjEmplTrans`	Projektstundenbuchungen
`ProjItemTrans`	Projektstunden und -ausgaben
`ProjInvoiceTable`	Projektrechnungskopf
`ProjInvoiceProjId`	Projektrechnungs-Zuordnung
`TSTimesheetTable`	Zeiterfassungserfassungskopf
`TSTimesheetLine`	Zeiterfassungszeilen

5.5. Human Resources (HR)

Wichtigste HR-Tabellen:

Tabelle	Inhalt
`HcmWorker`	Arbeitnehmerstamm (Mitarbeiter und Auftragnehmer)
`HcmEmployment`	Anstellungsverhältnisse
`HcmPosition`	Stellen
`HcmPositionWorkerAssignment`	Stellenzuordnung (Mitarbeiter-Stelle)
`HcmDepartment`	Abteilungen
`HcmJobTitle`	Jobbezeichnungen
`HrmAbsenceTable`	Abwesenheitserfassung
`PayrollWorkerPayStatement`	Lohnabrechnung (wenn Payroll-Modul aktiv)
`DirPerson`	Personenstamm (Global Address Book)
`DirPartyTable`	Party-Stamm (Basis fur Kunden, Lieferanten, Mitarbeiter)

Hinweis zum Global Address Book (GAB):

AX 2012 verwendet ein **Global Address Book** fur alle "Parties" (Personen und Organisationen):

- `DirPartyTable`: Basis-Stamm fur alle Parteien
- `DirPerson`: Personen-Erweiterung

- `DirOrganization`: Organisations-Erweiterung
- `LogisticsPostalAddress`: Postadressen
- `LogisticsElectronicAddress`: elektronische Adressen (Email, Telefon, Web)

Fast jede Entitat (Kunde, Lieferant, Mitarbeiter, Kontakt) ist uber eine `Party`-Referenz mit dem GAB verknupft.

5.6. Steuern (Tax)

Wichtigste Steuer-Tabellen:

Tabelle	Inhalt
`TaxTable`	Mehrwertsteuer-Codes
`TaxGroupData`	Steuergruppen-Zuordnungen
`TaxOnItem`	Artikel-Steuergruppen
`TaxTrans`	gebuchte Steuertransaktionen
`TaxGroupHeading`	Steuergruppen
`ExchangeRate`	Wechselkurse
`ExchangeRateType`	Wechselkurstypen

---

6 - Security-Modell und Benutzerrechte

6.1. Rollenbasiertes Sicherheitsmodell (RBAC)

AX 2012 fuhrte ein vollstandig neues **Role-Based Access Control (RBAC)** Modell ein, das das alte Gruppenmodell der Vorganger vollstandig ersetzte.

Hierarchie:

```
User
  → SecurityRole (Rolle)
      → SecurityDuty (Pflicht)
          → SecurityPrivilege (Privileg)
              → SecurableObject (Berechtigungsobjekt: Tabelle, Form, Report, ...)
```

Wichtigste Security-Tabellen:

Tabelle	Inhalt
`SecurityRole`	Rollendefinitionen
`SecurityUserRole`	Benutzer-Rollen-Zuordnungen
`SecurityDuty`	Pflichtendefinitionen
`SecurityRoleDutyExplode`	Rollen-Pflichten-Exploded-View
`SecurityPrivilege`	Privileg-Definitionen
`SecurityTask`	Aufgaben (ubergeordnete Kategorie)
`UserInfo`	AX-Benutzer
`SecurityDataPolicy`	Datensicherheitsrichtlinien (Datenebene)

6.2. Datensicherheitsrichtlinien (Record-Level Security)

AX 2012 unterstutzt **Record-Level Security** uber **Security Data Policies**:

- eine Policy definiert eine **XDS Query** (X++ Data Security Query)
- die Query filtert Tabellenzugriffe auf Datensatzebene
- ein Benutzer sieht nur Datensatze, fur die die Policy erfullt ist

Beispiel: Ein Vertriebsbenutzer sieht nur Aufträge seines eigenen Vertriebsgebiets — auch wenn er theoretisch Zugriff auf die `SalesTable` hat.

Implikation fur direkten SQL-Zugriff: Record-Level-Security-Policies wirken nur auf AOS-Ebene. Direkter SQL-Zugriff umgeht alle AX-Sicherheitsregeln vollstandig.

6.3. Benutzer und Session-Management

Technische Details:

- AX-Benutzer sind in `UserInfo` gespeichert
- AX-Benutzer-IDs sind maximal 8 Zeichen lang (`nvarchar(8)`)
- Authentifizierung erfolgt uber **Active Directory** (Windows-Authentifizierung) oder **Claims-Based Authentication** (fur Enterprise Portal)
- Sessions werden auf dem AOS verwaltet, nicht in der Datenbank

---

7 - Integration: AIF, Services und Schnittstellen

7.1. Application Integration Framework (AIF)

Das **Application Integration Framework** ist das native Integrationsframework von AX 2012. Es ermoglicht den Datenaustausch mit externen Systemen uber verschiedene Kanale.

AIF-Komponenten:

- **Services**: WCF-basierte Web Services (Standard, Custom, Wizard)
- **Ports**: Inbound- und Outbound-Integrationskanale
- **Pipelines**: Transformations- und Routinglogik
- **Adapters**: Anbindung an Transportsysteme (File, HTTP, MSMQ, ...)

AIF-Transporte:

Transport	Beschreibung
File System Adapter	Lesen/Schreiben von XML-Dateien
HTTP Adapter	HTTP/S-basierte Kommunikation
NetTcp Adapter	WCF NetTcp-Protokoll
MSMQ Adapter	Microsoft Message Queuing

7.2. WCF Services

AX 2012 stellt alle Geschaftsobjekte als **WCF-Services** bereit. Es gibt drei Typen:

Document Services:

- CRUD-Operationen auf AX-Entitaten
- stark typisierte XML-Schema-basierte Messages
- Beispiel: `SalesSalesOrderService` fur Verkaufsauftrags-CRUD

Custom Services:

- selbst entwickelte Services in X++
- vollige Freiheit im Daten- und Methodendesign
- Beispiel: eigener Lookup-Service fur externe Systeme

Query Services:

- ad hoc Abfragen uber AX-Queries
- kein Programmieraufwand, aber eingeschrankte Leistung

7.3. .NET Business Connector

Der **.NET Business Connector** ermoglicht direkten .NET-Code-Zugriff auf AX:

```
using Microsoft.Dynamics.BusinessConnectorNet;

var axBC = new Axapta();
axBC.Logon(null, null, null, null);

// Tabellenzugriff
var custTable = axBC.CreateAxaptaRecord("CustTable");
axBC.CallStaticRecordMethod("CustTable", "find", ref custTable, "10001", false);

Console.WriteLine(custTable.get_Field("Name").ToString());
axBC.Logoff();
```

Wichtig: Der .NET Business Connector unterliegt denselben AX-Sicherheitsregeln wie der normale Client.

7.4. ODBC und direkte SQL-Verbindungen

Fur Reporting, Analytics und Datenextraktion ist der direkte SQL-Server-Zugriff die haufigste und performanteste Option:

- ODBC/OLEDB-Verbindung direkt auf die AX SQL-Datenbank
- kein AOS-Overhead, keine AX-Lizenzkosten fur Lesezugriff
- keine AX-Sicherheitsregeln (Record-Level-Security wirkt nicht!)
- voller SQL-Zugriff auf alle Tabellen und Felder

Best Practice fur Analytics:

- eigenen **Read-Replica** oder **Leseberechtigung** auf der AX-Datenbank einrichten
- nie Produktions-AX-Datenbank direkt fur schwere Analyseabfragen nutzen
- bei vollstandiger Datenextraktion: Always-On-Readable-Secondary oder Snapshot-Isolation verwenden

---

8 - Reporting: SSRS, Management Reporter und OLAP

8.1. SQL Server Reporting Services (SSRS)

AX 2012 verwendet **SSRS** als native Berichtsplattform. Berichte werden im AOT als `Report`-Objekte definiert und beim Deployment auf einen SSRS-Server deployed.

Technische Details:

- AX-Reports sind **RDL-basiert** (Report Definition Language)
- Datenzugriff erfolgt uber **AX Query Objects** oder **AX Business Logic** (nicht direktes SQL)
- Deployment: AOT-Report → AX-Compile → SSRS-Report-Server
- Berichte werden im AX-Client ausgefuhrt und im PDF/Excel/Word-Format exportiert

Wichtige SSRS-Berichte in Standard-AX:

- Debitorenaltersliste (Aged Balances)
- Kreditoren-Zahlungsvorschlag
- Lagerbewertung
- Produktionskostenanalyse
- Projektgewinn- und -verlust

8.2. Management Reporter (Financial Reporter)

Management Reporter ist das Finanzberichts-Werkzeug, das fur Geschaftsberichte (GuV, Bilanz, Cashflow) optimiert ist.

- Zugriff auf AX-Finanzdaten uber eine eigene ODBC-Verbindung
- GUI-basierter Report-Designer (keine SSRS-Kenntnisse notig)
- unterstutzt Finanzdimensionen fur Auswertungen nach Kostenstelle, Profit Center, etc.
- kann Budgets gegen Istwerte gegenuberstellen

8.3. OLAP-Cubes (Analysis Services)

AX 2012 enthalt **fertige OLAP-Cubes** fur SQL Server Analysis Services:

Cube	Inhalt
Expense management	Reisekostenauswertungen
Ledger	Hauptbuch-Cube
Sales	Vertriebsanalyse
Purchase	Einkaufsanalyse
Inventory value	Lagerbewertung
Production	Fertigungsanalyse
Budget	Budgetkontrolle
Project accounting	Projektabrechnung

Technische Details:

- Cubes werden uber den **Microsoft Dynamics AX Cube Setup Wizard** deployt
- Datenquelle sind Perspective-Views auf der AX SQL-Datenbank

- Cubes werden per SSAS-Processing aktualisiert (typischerweise nachts)
- Power BI Desktop kann direkt auf Analysis Services Cubes zugreifen

---

9 - Performance und Datenbankoptimierung

9.1. Datenbankgrosse und Wachstum

AX-Datenbanken wachsen erheblich:

- **InventTrans**: in grossen Installationen oft 100+ Millionen Zeilen
- **GeneralJournalAccountEntry**: ebenfalls sehr gross nach jahrelangem Betrieb
- **LedgerTrans**: kann bei langen Betriebszeiten sehr gross werden
- **SalesLine / PurchLine**: wachsen kontinuierlich, werden aber selten archiviert

Typische Datenbankgrosse:

- kleine Installation: 50-200 GB
- mittlere Installation: 200 GB - 2 TB
- grosse Installation: 2-10+ TB

9.2. Indexstruktur

AX-Indizes werden im AOT definiert und auf SQL-Server synchronisiert. Typische Index-Typen:

Typ	Verwendung
Clustered Index	meist auf RecId, je Tabelle einer
Unique Index	naturliche Schlussel (z.B. AccountNum)
Non-Clustered Index	Suchindex fur haufige Abfragen

Praxishinweis: Bei direktem SQL-Zugriff fur Analytics immer **Covering Indexes** nutzen oder sicherstellen, dass der WHERE-Predicate einen vorhandenen Index nutzt. `EXPLAIN PLAN` (SQL Server: Query Execution Plan) ist das wichtigste Debugging-Werkzeug.

9.3. Blocking und Locking

AX 2012 verwendet auf Datenbankebene **Row-Level Locking** (SQL Server Standard). Typische Lock-Probleme:

- lange Batch-Jobs sperren Tabellenzeilen
- gleichzeitiger Zugriff auf `SystemSequences` (RecId-Vergabe) kann Blockierungen verursachen
- grosse Updates ohne korrekte Transaction-Boundaries

Wichtig fur Analytics: Bei schweren Leseabfragen auf der Produktionsdatenbank `WITH (NOLOCK)` oder `READ UNCOMMITTED` Isolation Level nutzen — oder besser eine Read-Replica verwenden.

9.4. Tabellenpartitionierung (Database-Level)

Nicht zu verwechseln mit AX Data Partitions (logische Trennung von Instanzen): **SQL Server Table Partitioning** kann fur grosse AX-Tabellen eingesetzt werden, um:

- Archivierung zu erleichtern

- Backup-Performance zu verbessern
- Index-Verwaltung zu beschleunigen

In Standard-AX ist keine SQL-Table-Partitionierung vorkonfiguriert — das ist eine DBA-Aufgabe.

---

10 - Migration nach Dynamics 365 Finance & Operations

10.1. Warum migrieren?

Microsoft hat den Extended Support fur AX 2012 beendet. Grunde fur eine Migration nach **D365 Finance & Operations**:

Für die fachliche Zielarchitektur ist wichtig, dass Dynamics 365 heute nicht nur aus Finance & Operations besteht. Die moderne Microsoft-Business-Applications-Welt umfasst unter anderem Finance, Supply Chain Management, Commerce, Project Operations, Sales, Customer Service, Field Service, Dataverse, Power Platform, Power BI und Azure-Integrationsdienste. Der Artikel Dynamics 365: ERP, CRM, Power Platform und Azure im Microsoft-Oekosystem ordnet diese Landschaft ausführlich ein.

- **Security**: keine weiteren Sicherheits-Patches fur AX 2012
- **Cloud-Benefits**: automatische Updates, keine On-Premises-Infrastruktur
- **Neue Funktionen**: Power Platform-Integration, Copilot, AI Builder
- **Ecosystem**: bessere Integration mit Fabric, Power BI, Azure
- **Lizenzmodell**: Subscription statt Einmallizenz

10.2. Migration-Pfade

Microsoft bietet zwei Hauptpfade:

Pfad 1 — Greenfield-Implementierung:

- Neuimplementierung in D365 FO
- Datenmigration von AX zu D365
- Business-Prozesse werden neu modelliert
- Empfohlen wenn: viele Anpassungen, veraltete Prozesse, schlechte Datenqualitat

Pfad 2 — Lift-and-Shift / Upgrade:

- technisches Upgrade der AX-Codebase nach D365
- automatisiertes Code-Upgrade-Tool (Dynamics 365 Migration-Tool)
- Empfohlen wenn: wenige Anpassungen, gute Datenqualitat, enge Timelines

10.3. Datenmigration — Technische Aspekte

Bei der Datenmigration von AX zu D365 FO sind die wichtigsten Werkzeuge:

Data Management Framework (DMF):

- AX 2012 und D365 FO haben beide ein DMF
- Import/Export uber definierte Entitaten (Data Entities)
- unterstutzt CSV, Excel, XML, JSON

Dynamics 365 Migration-Tool:

- von Microsoft bereitgestelltes Tool fur Code-Migration (X++ zu X++ fur D365)
- analysiert Customizations und schlagt Migrationspfade vor

Handmigration via SQL:

- fur grosse historische Datenmengen oft die einzige praktikable Option
- direkte SQL-Extraktion aus AX, Transformation und Laden via DMF in D365

10.4. Typische Herausforderungen bei der Datenmigration

Herausforderung	Ursache	Lösungsansatz
RecId-Konflikte	D365 vergibt neue RecIds	Mapping-Tabelle, kein RecId-Transfer
DataAreaId-Anderung	unterschiedliche Company-Codes	Mapping und Transformation
Pflichtfeld-Validierungen	D365 erzwingt Felder, die AX 2012 nicht hatte	Datensanierung vor Migration
Anpassungs-Tabellen	Custom Tables aus AX existieren in D365 nicht	Code-Migration und Daten-Migration koordinieren
OPOS/Offene Posten	offene Posten mussen korrekt uberfuhrt werden	Stichtags-Migration
Bestandsmigrierung	InventTrans-Massen ohne Kontext	Einstandsmengen-Migration
Finanzdaten	gebuchte LedgerTrans ohne Voucher	Opening-Balance-Migration

---

11 - AX 2012 als Datenquelle fur moderne Analytics

11.1. Extraktion nach Microsoft Fabric

AX 2012 ist fur viele Unternehmen der primäre Datenlieferant fur **Microsoft Fabric**. Die wichtigsten Extraktionsmuster:

Muster 1 — Direkter SQL-Zugriff via Copy Job:

Fur SQL-Server-Quellen mit einem On-Premises-Daten-Gateway:

- Fabric Copy Job mit SQL-Server-Konnektor
- Incremental Copy auf `ModifiedDateTime` fur Delta-Ladung
- Bronze-Landing in Fabric Lakehouse
- Silver-Merge per Notebook

Muster 2 — SSAS-Cube-Integration:

Wenn AX-Cubes bereits vorhanden sind:

- Power BI Desktop direkt auf Analysis Services Cubes
- Import-Mode oder DirectQuery je nach Größe
- keine weitere Transformation notwendig für Standardberichte

Muster 3 — AIF-/Service-basierte Extraktion:

Fur spezifische Echtzeit-Integrationen:

- AX AIF Document Services als API
- Fabric Data Pipeline mit HTTP-Connector
- aufwändiger, aber ermöglicht delta-basierte Extraktion ohne direkten SQL-Zugriff

11.2. Die wichtigsten AX-Tabellen fur Analytics

Fur eine typische AX-Analytics-Architektur sind diese Tabellen die wichtigsten Quellen:

Finanzen:

Tabelle	Priorität	Zeilen (typisch)
`GeneralJournalAccountEntry`	hoch	sehr viele
`LedgerTrans`	hoch	viele (historisch)
`CustTrans`	hoch	viele
`VendTrans`	hoch	viele
`CustInvoiceJour`	hoch	mittel
`VendInvoiceJour`	hoch	mittel
`TaxTrans`	mittel	mittel

Logistik:

Tabelle	Priorität	Zeilen (typisch)
`InventTrans`	hoch	sehr viele
`SalesTable`	hoch	mittel
`SalesLine`	hoch	viele
`PurchTable`	hoch	mittel
`PurchLine`	hoch	viele
`InventSum`	mittel	mittel (aggregiert)

Stammdaten:

Tabelle	Priorität	Zeilen (typisch)
`CustTable`	hoch	wenig bis mittel
`VendTable`	hoch	wenig bis mittel
`InventTable`	hoch	mittel
`EcoResProduct`	hoch	mittel
`DirPartyTable`	mittel	mittel
`MainAccount`	mittel	wenig
`DataArea`	mittel	wenig

11.3. Empfohlene Ladestrategie je Tabelle

Tabelle	Ladestrategie	Merge-Key	Inkrementelle Spalte
`CustTable`	Incremental + Merge	`RecId`	`ModifiedDateTime`

Tabelle	Ladestrategie	Merge-Key	Inkrementelle Spalte
`VendTable`	Incremental + Merge	`RecId`	`ModifiedDateTime`
`InventTable`	Incremental + Merge	`RecId`	`ModifiedDateTime`
`SalesTable`	Incremental + Merge	`RecId`	`ModifiedDateTime`
`SalesLine`	Incremental + Merge	`RecId`	`ModifiedDateTime`
`PurchTable`	Incremental + Merge	`RecId`	`ModifiedDateTime`
`PurchLine`	Incremental + Merge	`RecId`	`ModifiedDateTime`
`InventTrans`	Incremental + Append	`RecId`	`DatePhysical` oder `ModifiedDateTime`
`CustTrans`	Incremental + Append	`RecId`	`TransDate` oder `ModifiedDateTime`
`VendTrans`	Incremental + Append	`RecId`	`TransDate` oder `ModifiedDateTime`
`GeneralJournalAccountEntry`	Incremental + Append	`RecId`	`AccountingDate`
`CustInvoiceJour`	Incremental + Merge	`RecId`	`ModifiedDateTime`
`MainAccount`	Full + Replace	`RecId`	—
`DataArea`	Full + Replace	`RecId`	—
`InventDim`	Incremental + Append	`RecId`	— (keine Änderungsspalte)

Besonderheiten:

- `InventDim` hat keine verlässliche Änderungsspalte — entweder Full Load oder Append ohne Delta
- `InventTrans` wird selten nachträglich geändert — Append ist meistens korrekt
- **Transaktions-Tabellen** (CustTrans, VendTrans, LedgerTrans): Status-Änderungen (z.B. Ausgleich) können nachträgliche Updates erzeugen — Merge bevorzugen

11.4. SQL-Abfrage-Beispiele für Analytics

Debitorenaltersliste:

```
SELECT
    ct.AccountNum,
    c.Name,
    ct.Invoice,
    ct.TransDate,
    ct.Due,
    ct.AmountMST,
    ct.AmountCur,
    ct.CurrencyCode,
    DATEDIFF(DAY, ct.Due, GETDATE()) AS DaysOverdue,
    CASE
        WHEN DATEDIFF(DAY, ct.Due, GETDATE()) <= 0  THEN 'Current'
        WHEN DATEDIFF(DAY, ct.Due, GETDATE()) <= 30 THEN '1-30 Days'
        WHEN DATEDIFF(DAY, ct.Due, GETDATE()) <= 60 THEN '31-60 Days'
        WHEN DATEDIFF(DAY, ct.Due, GETDATE()) <= 90 THEN '61-90 Days'
        ELSE 'Over 90 Days'
    END AS AgingBucket
FROM CustTrans ct
INNER JOIN CustTable c
    ON c.AccountNum = ct.AccountNum
   AND c.DataAreaId = ct.DataAreaId
   AND c.Partition  = ct.Partition
WHERE ct.DataAreaId = 'USMF'
  AND ct.Partition  = 5637144576
```

```
  AND ct.Open_      = 1
ORDER BY ct.AccountNum, ct.Due;
```

Lagerbewertung je Artikel:

```
SELECT
    it.ItemId,
    ep.Name AS ProductName,
    ig.Name AS ItemGroupName,
    SUM(isum.PhysicalInventory)     AS PhysicalQty,
    SUM(isum.FinancialInventory)    AS FinancialQty,
    SUM(isum.PhysicalValue)         AS PhysicalValue,
    SUM(isum.FinancialValue)        AS FinancialValue
FROM InventSum isum
INNER JOIN InventTable it
    ON it.ItemId    = isum.ItemId
   AND it.DataAreaId = isum.DataAreaId
   AND it.Partition  = isum.Partition
LEFT JOIN EcoResProduct ep
    ON ep.RecId = it.Product
   AND ep.Partition = it.Partition
LEFT JOIN InventItemGroup ig
    ON ig.ItemGroupId = it.ItemGroupId
   AND ig.DataAreaId  = it.DataAreaId
   AND ig.Partition   = it.Partition
WHERE isum.DataAreaId = 'USMF'
  AND isum.Partition  = 5637144576
  AND (isum.PhysicalInventory <> 0 OR isum.FinancialInventory <> 0)
GROUP BY it.ItemId, ep.Name, ig.Name
ORDER BY it.ItemId;
```

Umsatz nach Kunde und Monat:

```
SELECT
    cij.InvoiceAccount,
    ct.Name,
    YEAR(cij.InvoiceDate) AS InvoiceYear,
    MONTH(cij.InvoiceDate) AS InvoiceMonth,
    SUM(cij.SumTax)     AS TaxAmount,
    SUM(cij.SumMarkup)  AS FreightAmount,
    SUM(cit.LineAmount) AS NetAmount,
    SUM(cij.SumTax + cit.LineAmount) AS GrossAmount
FROM CustInvoiceJour cij
INNER JOIN CustInvoiceTrans cit
    ON cit.InvoiceId   = cij.InvoiceId
   AND cit.DataAreaId  = cij.DataAreaId
   AND cit.Partition   = cij.Partition
LEFT JOIN CustTable ct
    ON ct.AccountNum  = cij.InvoiceAccount
   AND ct.DataAreaId  = cij.DataAreaId
   AND ct.Partition   = cij.Partition
WHERE cij.DataAreaId  = 'USMF'
  AND cij.Partition   = 5637144576
  AND cij.InvoiceDate >= '2025-01-01'
GROUP BY cij.InvoiceAccount, ct.Name,
         YEAR(cij.InvoiceDate), MONTH(cij.InvoiceDate)
ORDER BY cij.InvoiceAccount, InvoiceYear, InvoiceMonth;
```

11.5. Wichtige Datenbankhinweise fur Analytics-Entwickler

DataAreaId und Partition immer filtern:

```
-- IMMER so:
SELECT ... FROM CustTable WHERE DataAreaId = 'USMF' AND Partition = 5637144576
```

```
-- NIE so (liest alle Companies und Partitionen):
SELECT ... FROM CustTable
```

Partition-ID herausfinden:

```
-- Verfugbare Partitionen und Companies
SELECT ps.PartitionKey, da.Id, da.Name, da.RecId
FROM PartitionSys ps
CROSS JOIN DataArea da
WHERE da.Partition = ps.RecId
ORDER BY ps.PartitionKey, da.Id;
```

ModifiedDateTime ist UTC:

AX speichert `ModifiedDateTime` immer in UTC. Bei der Anzeige in lokaler Zeitzone muss konvertiert werden:

```
-- UTC nach MEZ (vereinfacht, ohne Sommerzeit):
DATEADD(HOUR, 1, ModifiedDateTime) AS ModifiedDateTime_MEZ
```

Numerische Felder in AX:

AX 2012 verwendet fur Geldbeträge den Typ `numeric(32,6)` — sehr hohe Prazision, oft unnötig. In Fabric kann zu `DECIMAL(18,4)` oder `DECIMAL(18,2)` konvertiert werden.

Beachten Sie den Unterschied zwischen physischen und finanziellen Buchungen:

- `DatePhysical` / `PhysicalInventory` / `PhysicalValue`: physische Warenbewegung (Lieferschein)
- `DateFinancial` / `FinancialInventory` / `FinancialValue`: finanzielle Buchung (Rechnung)
- fur Bestandsbewertungen immer den finanziellen Stand verwenden
- fur operative Reports oft der physische Stand relevanter

---

12 - Fazit

Microsoft Dynamics AX 2012 ist trotz seines Alters ein technisch ausgereiftes und ausserst leistungsfahiges ERP-System. Fur IT-Architekten und Datenentwickler ist es heute vor allem als:

1. **System of Record** in laufenden Produktionsinstanzen
2. **Datenquelle** fur moderne Cloud-Analytics-Plattformen
3. **Migrationsprojekt-Ursprung** fur D365 FO-Migrationen

In allen drei Rollen ist ein tiefes Verstandnis der Architektur — besonders des Datenbankschemas, der technischen Pflichtfelder (`RecId`, `DataAreaId`, `Partition`, `ModifiedDateTime`) und der Modulstruktur — unverzichtbar.

Die wichtigsten Takeaways:

- `RecId` = technischer Primärschlüssel, pro Tabelle eindeutig
- `DataAreaId` = Company-Filter, IMMER mitfuhren
- `Partition` = Instanz-Filter (ab R2), mitfuhren wenn mehrere Partitionen existieren
- `ModifiedDateTime` = Änderungsspalte fur Incremental Load, UTC
- **AX-Sicherheit** wirkt nur auf AOS-Ebene — direkter SQL-Zugriff hat keine Sicherheitsfilter
- fur Analytics ist **direkter SQL-Zugriff** auf eine Read-Replica der empfohlene Weg

---

13 - Literatur und Ressourcen

Bücher

- Microsoft Dynamics AX 2012 R3 Development Cookbook — praktische Kochbuch-Sammlung fur AX-Entwickler
- Microsoft Dynamics AX 2012 R3 Reporting Cookbook — SSRS-Reporting in AX
- Grundkurs Microsoft Dynamics AX: Die Business Lösung von Microsoft in der Version AX 2012 — deutschsprachige Einstiegslekture
- Inside Microsoft Dynamics AX 2012 — technische Tiefe, empfohlen fur Entwickler und Architekten
- Learning MS Dynamics AX 2012 Programming — Einstieg in X++ Programmierung

Online-Ressourcen

- Microsoft Docs: Dynamics AX 2012 — offizielle Microsoft-Dokumentation (archiviert)
- AX 2012 System Architecture TechNet — Architektur-Dokumentation
- Lifecycle Services (LCS) — Projekt-Management und Diagnose fur D365 FO (Nachfolge-Platform)

Migration nach D365 FO

- Dynamics 365 Finance Migration Overview
- Data Migration Framework for D365

Über den Autor

Erhard Rainer berät kleine und mittlere Unternehmen in Österreich zu Business Intelligence, Machine Learning und KI-Agenten. Er hat 15 Jahre Erfahrung in Data Analytics, davon 10 Jahre in der Leitung von Teams und Departments in Konzernen, und ist Dozent für betriebswirtschaftliche und Data-Analytics-Fächer.

Sprechen Sie mit mir über Ihre Daten: Das Erstgespräch ist kostenlos.

https://www.thinkbigger.at/contact.html#anfrage

© 2026 Think BIGGER – Erhard Rainer. Dieses PDF gibt den Stand des Online-Artikels vom 07.07.2026 wieder; maßgeblich ist die Online-Fassung unter https://www.thinkbigger.at/wissen/dynamics-ax-2012/