Power BIのデータモデル設計入門

速く、正しく、壊れにくいレポートをつくる



はじめに：Power BIの品質はデータモデルで決まる

Power BIの成果を左右するのはグラフの作り方ではなくデータモデル設計である。設計が不適切な場合、数値の不一致と処理遅延が同時に発生し、レポートの信頼性そのものが失われる。

現状

Excelの集計作業をそのままPower BIに持ち込み、1枚の大きな表からレポートを作成している。

グラフは表示できるが、集計の前提が担当者ごとに異なる状態が残る。

重要課題

・合計値が想定より大きくなる
・ExcelとPower BIで結果が一致しない
・マスタ変更時にデータが不整合となる
・データ量の増加で更新・表示が遅くなる

本資料で学べること

① テーブルの役割（ファクト／ディメンション）
② スタースキーマ
③ リレーションシップ
④ メジャーと計算列の使い分け

期待される状態

集計ロジックが一元化され、誰が見ても同じ数値になる。

新しい分析軸の追加やデータ量の増加にも、構造を壊さず対応できる。

本資料の前提：ビジュアル作成の前に、テーブルの役割分担・粒度・リレーションシップを決める。この順序を守るだけで、数値の不一致と処理遅延の大半は設計段階で回避できる。

事実：Power BIはPower Query／データモデル／ビジュアルの3層で構成される | 解釈：不具合の多くはビジュアルではなくモデル層に起因する | 提案：設計順序を「モデル→ビジュアル」に固定する

データモデルとは何か

データモデルは、複数のテーブルを分析しやすい形に整理する仕組みである。整形（Power Query）・構造と集計定義（データモデル）・表現（ビジュアル）を分離することで、集計ロジックの統一と更新の自動化が同時に実現する。

Power BIの3層構造と役割

① Power Query（整形）

接続・列の取捨選択・型変換・表記の統一・不要行の除去。
「読み込む前にきれいにする」層。

② データモデル（構造・集計）

テーブルの役割分担、リレーションシップ、日付テーブル、メジャー定義。
数値の正しさを担保する中核層。

③ ビジュアル（表現）

グラフ・表・スライサーによる可視化。
モデルが正しければ、表現の変更は低コストで済む。

Excelの1枚表とPower BIのデータモデルの違い

観点	Excelの1枚表	Power BIのデータモデル
構造	実績と属性を同一表に混在	実績と分析軸をテーブルで分離
集計定義	シート・数式ごとに個別	メジャーとして一元管理
分析軸の追加	列追加と再集計が必要	マスタ追加とリレーションで対応
更新	手作業の貼り付けが中心	更新処理で自動反映

データモデルを設計するメリット

集計ロジックの統一

売上・粗利の定義をメジャーに集約し、レポート間の差異をなくす。

レポートの再利用

同一モデル上に部門別・用途別のレポートを追加できる。

更新作業の自動化

手作業の転記を排除し、更新のたびの再作成を不要にする。

データ量増加への対応

重複を排した構造により、行数増加でも性能を維持しやすい。

事実：Power BIは整形・モデル・表現の3層に処理が分かれる | 解釈：層をまたいで同じ処理を書くと定義が分散し不一致の原因となる | 提案：整形はPower Query、集計はメジャーに責務を固定する

よくある失敗：すべてを1枚の表にまとめる

売上・顧客・商品・担当者を1表に集約すると、属性値が取引の行数だけ重複する。容量と処理負荷が増えるだけでなく、マスタ変更時に修正漏れが生じ、データ不整合が構造的に発生する。

As-Is：1枚表（フラットテーブル）の例

取引日	顧客名	顧客住所	商品名	分類	担当者	売上金額
04-01	A商事	東京都…	製品X	機器	山田	120,000
04-02	A商事	東京都…	製品X	機器	山田	80,000
04-02	A商事	東京都…	製品Y	消耗品	山田	45,000
04-03	A商事	東京都…	製品X	機器	鈴木	120,000

赤字部分は同一の属性値の重複。顧客住所を1件変更するだけで、該当する全取引行の修正が必要となり、修正漏れがそのまま集計誤差になる。

ExcelではてもPower BIでは管理しにくい理由

・Excelは「人が見る一覧表」、Power BIは「機械が集計する構造」を前提とする
・1枚表では属性の重複により列のカーディナリティが高くなり、圧縮効率が落ちる
・分析軸を追加するたびに列を増やす運用になり、表構造が肥大化して保守できない
・スライサーで使う分類項目が実績行に埋め込まれるため、値の一覧を正しく管理できない
・属性の正しい値が1か所に存在しないため、集計結果の検証が困難になる

1枚表がもたらす4つの問題

① 属性値の重複

顧客名・商品名・分類が取引の行数だけ繰り返し保持される。

② 容量と処理負荷の増加

文字列列の重複がファイル容量を押し上げ、更新・表示の待ち時間が伸びる。

③ マスタ変更時の不整合

名称変更・分類変更が全行に反映されず、同一顧客が複数表記に分裂する。

④ 集計結果への影響

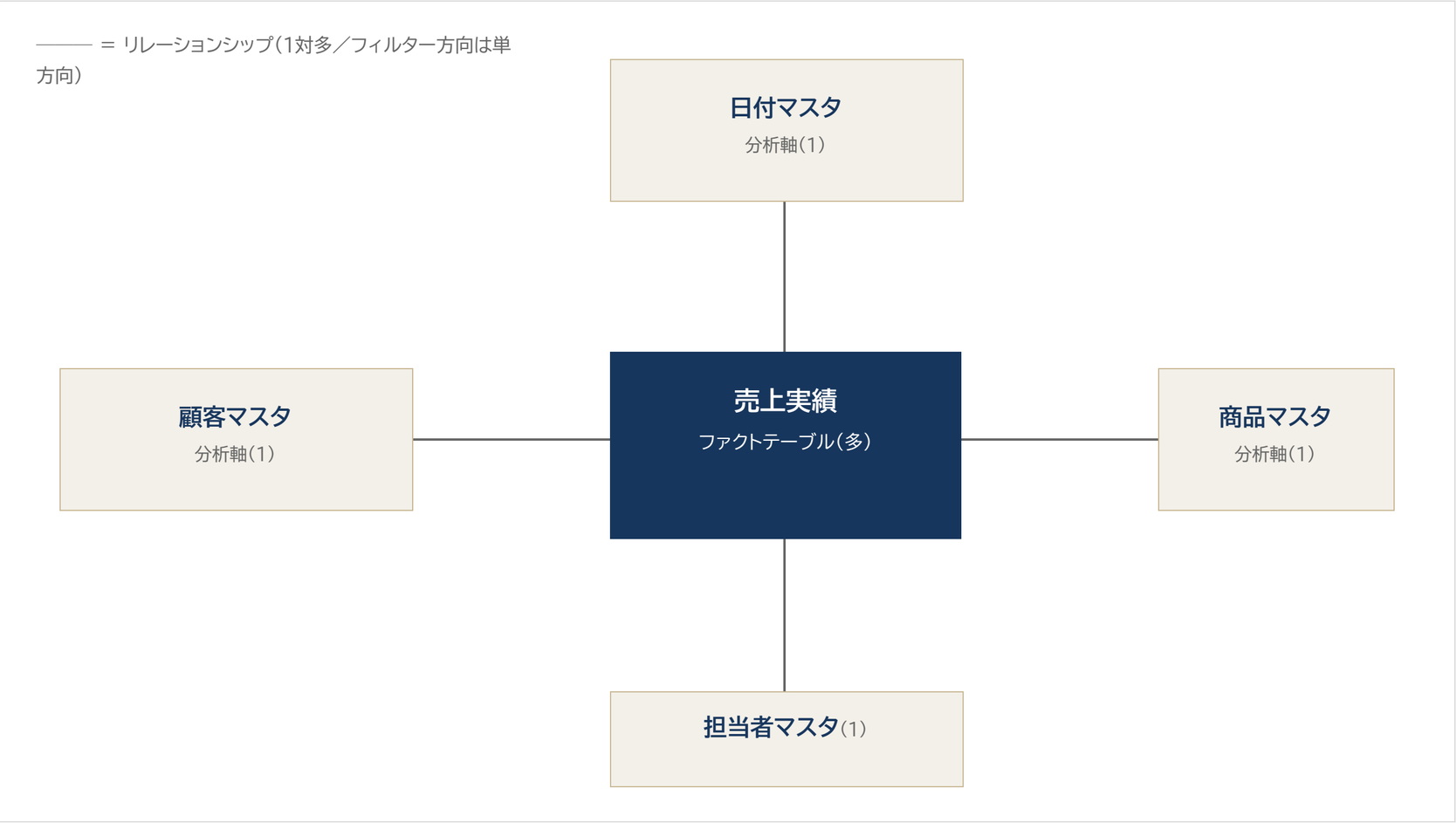
表記ゆれが分析軸を分断し、顧客別・商品別の合計が実態と一致しなくなる。

事実：1枚表では属性値が取引行数だけ重複保持される | 解釈：重複は容量増加ではなく「正しい値の所在が不明になる」ことが本質的な問題 | 提案：属性はマスタ側に1件のみ保持し、実績側はIDのみを持つ

基本となるスタースキーマ

中央に実績（ファクト）、周囲に分析軸（ディメンション）を配置する星形構造を基本形とする。単純な構造ほど集計の解釈が一意に定まり、保守と拡張が容易になる。

売上分析モデルの構成例



スタースキーマの要点

構造の原則

- ファクトは中央に1つ、数値と外部キーのみを持つ
- ディメンションは周囲に配置し、IDを一意にする
- 関係は必ずマスタ（1）→実績（多）の向き
- フィルターはディメンションから流し、ファクトを絞り込む
- マスタ同士を直接つながない

得られる効果

- 集計の意味が一意に定まり、合計値の検証が容易になる
- 分析軸の追加はマスタ1件の追加で完了する
- 属性の重複が排除され、更新・表示が軽くなる
- 第三者が構造を短時間で理解できる

事実：スタースキーマは中央のファクトと周囲のディメンションで構成される | 解釈：構造の単純さが集計結果の説明可能性を担保する | 提案：例外設計を導入する前に、まず星形で成立するかを検討する

ファクトテーブルの設計

ファクトテーブルは分析対象の実績データを保持し、外部キーと数値のみで構成する。設計の出発点は「1行が何を表すか」（粒度）の定義であり、粒度が曖昧なままでは合計値の正しさを検証できない。

主な項目構成（売上実績テーブル）

項目	区分	設計上の要点
取引日	外部キー	日付テーブルと関連付ける。時刻は分析に必要な場合のみ保持
顧客ID	外部キー	顧客名ではなくIDを保持。マスタ側に名称を置く
商品ID	外部キー	分類・単価などの属性はマスタで管理する
担当者ID	外部キー	組織変更に備え、担当者名を実績側に持たせない
数量	数値	整数型。合計・平均の対象となる
売上金額	数値	粒度に整合した金額を保持。比率は行に持たず、メジャーで計算
原価	数値	粗利の算出に使用。売上金額と同一粒度でそろえる

原則：ファクトテーブルには「IDと数値」だけを置く。名称・住所・分類などの説明項目を持たせると、1枚表と同じ重複問題が再発する。

粒度（1行が表す単位）の考え方

① 日次集計単位

1行＝1日×商品×顧客の合計。行数は少なく高速だが、伝票単位の分析はできない。
用途：月次推移・全社KPIの把握

② 伝票単位

1行＝1伝票。受注件数や平均単価の分析に適するが、明細別の内訳は追えない。
用途：取引件数・受注単価の分析

③ 商品明細単位（推奨の起点）

1行＝伝票内の1明細。最も細かい粒度のため、上位の集計はすべて再現できる。
用途：商品別・顧客別・担当者別の全分析
留意：細かいほど行数が増えるため、必要な粒度の下限を業務要件から決める。

事実：ファクトの粒度は集計可能な最小単位を規定する | 解釈：粒度が異なる数値を同一テーブルに混在させると合計が二重計上となる | 提案：粒度を1文で言語化し、モデル上の説明欄に明記する

ディメンションテーブルの設計

ディメンションテーブルは分析軸として使うマスタであり、IDの一意性が品質の前提となる。表示名称と分類項目をマスタ側に集約し、ファクトテーブルに名称を重複させないことが原則である。

設計例：顧客マスタ／商品マスタ

テーブル	主キー（一意）	表示名称・分類項目
顧客マスタ	顧客ID	顧客名／業種／地域／顧客区分
商品マスタ	商品ID	商品名／カテゴリ／サブカテゴリ／標準単価
担当者マスタ	担当者ID	担当者名／所属部門／拠点
店舗・部門マスタ	店舗ID／部門ID	店舗名／部門名／エリア／管轄

表記ゆれ・重複の除去（Power Queryで前処理）

除去前（同一顧客が分裂）

C001／株式会社A商事
C002／（株）A商事
C003／A商事　株式会社
→ 顧客別売上が3つに分断される

除去後（一意に統合）

C001／株式会社A商事
（表記統一・前後空白除去・重複行削除）

→ 顧客別売上が実態と一致する

注意：IDが重複した状態ではPower BIが「1対多」のリレーションシップを作成できない。マスタ側の一意性確保は、リレーションシップ設定の前提条件である。

設計の5原則

① IDを一意にする

1件のID＝1行。重複・空白があれば読み込み前に解消する。

② 表示名称を持たせる

グラフの軸に表示する名称はマスタ側に1件だけ保持する。

③ 分類項目を整備する

業種・地域・カテゴリなど、集計の切り口となる列を用意する。

④ 表記ゆれ・重複を除去する

全角半角・空白・法人格表記を統一し、同一実体を1件に集約する。

⑤ 名称をファクトに重複させない

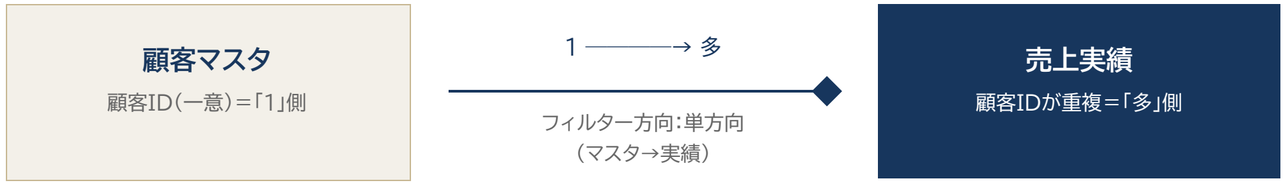
実績側はIDのみ。名称はリレーション経由で参照する。

事実：ディメンションの主キーが一意でない場合は1対多の関係を定義できない｜解釈：表記ゆれは技術問題ではなく業務上のマスタ運用の問題｜提案：マスタ整備の責任部署と更新手順を明文化する

リレーションシップの基本

リレーションシップは「マスタ側を1、実績側を多」とし、フィルター方向は原則単方向とする。多対多や双方向フィルターは集計結果の解釈を不確定にするため、代替手段を検討したうえで例外的に使用する。

基本形：1対多・単方向フィルター



設定時の確認事項

確認項目	推奨設定	確認しない場合に起きること
カーディナリティ	1対多（マスタ＝1）	合計値の二重計上
IDのデータ型	両側で統一（整数か文字列）	リレーションシップを作成できない
フィルター方向	単方向	絞り込み結果が予測できない
空白・不一致キー	事前に補完・除去	「空白」行に売上が集約される

安易に使用すべきでない設定

多対多のリレーションシップ

両側にキーの重複があるため、集計の対応関係が一意に決まらず合計が膨らむ。
対応：両者に共通する橋渡し用マスタ（ブリッジ）を作り、1対多に分解する。

双方向フィルター

フィルターが逆流し、想定外の絞り込みや循環参照を生む。
対応：まず単方向で構成し、必要な箇所のみDAX関数で一時的に方向を変える。

アクティブ／非アクティブな関係

2つのテーブル間に複数の関係を持たせる場合、常時有効なのは1本のみ（アクティブ／実線）。残りは非アクティブ（点線）となる。

例：売上実績と日付テーブルを「受注日」「出荷日」の2本で結ぶ場合、既定集計は受注日側のみに適用される。出荷日基準の集計は、非アクティブな関係を明示的に有効化するメジャーで定義する。

事実：2テーブル間でアクティブな関係は1本のみ｜解釈：双方向・多対多は「動くが説明できない集計」を生みやすい｜提案：既定は1対多・単方向とし、例外は理由を注記に残す

日付テーブルの作成

売上日の列だけでは月次・四半期・前年比の分析が成立しない。連続した日付を持つ独立した日付テーブルを作成し、「日付テーブルとしてマーク」することで、期間比較のすべてが同一の時間軸で扱えるようになる。

日付テーブルに持たせる項目

列	値の例	主な用途
日付(主キー)	2026/04/01	ファクトとの関連付け・日次推移
年	2026	年次比較・スライサー
月	4	月別集計・並び順の制御
年月	2026-04	月次推移グラフの軸
四半期	Q1	四半期集計・経営報告
曜日	水	曜日傾向の分析・稼働日判定
会計年度	FY2026	決算期に合わせた期間比較

連続性の要件: 分析対象期間の全日付を欠落なく登録する。取引が存在しない日を除外すると、期間比較の分母が変動し、前年同月比や移動平均が正しく算出されない。

必須操作: 作成した日付テーブルを「日付テーブルとしてマーク」する。これによりタイムインテリジェンス関数(前年同期・累計など)が正しく機能する。

複数の日付列がある場合の考え方

状況:売上日・受注日・出荷日が併存

日付テーブルは1つだが、ファクト側の日付列は複数。関係を複数本引くとアクティブは1本のみとなる。

方針A:主たる基準日を1つ決める(推奨)

売上計上の基準日(例:売上日)をアクティブな関係とし、他は非アクティブにする。別基準の集計は、その関係を有効化するメジャーとして個別に定義する。

方針B:役割別の日付テーブルを追加

受注日用・出荷日用の日付テーブルを別に用意し、それぞれ1対多・単方向で接続する。並べて比較する要件があるときに有効。

回避すべき対応: 基準日ごとにファクトを複製する。行数が倍増し、合計値が二重計上となる。

事実:タイムインテリジェンス関数は連続した日付テーブルを前提とする | 解釈:日付列の直接利用では期間比較の基準が保証されない | 提案:モデル作成の初期段階で日付テーブルを作成し、マーク設定まで完了させる

計算列とメジャーの使い分け

分類・ラベルなど行単位で固定する値は計算列、合計・比率・KPIなど選択条件に応じて変化する値はメジャーで定義する。集計を計算列で持つと、フィルターに反応せず数値の不一致を招く。

4つの観点による比較

項目	計算列	メジャー
計算タイミング	データ更新時	レポート表示時
主な用途	分類・ラベル・行単位計算	合計・比率・KPI
保存領域	使用する	原則として使用しない
フィルターへの反応	固定値	選択条件に応じて変化

計算列が適する例

・売上規模の区分ラベル付与
・氏名の結合、コードから区分名の変換
・グラフの軸やスライサーに使う分類列
・並び順制御用の数値列

メジャーが適する例

・売上金額・粗利益などの合計
・粗利益率・達成率などの比率
・前年同月比、累計、構成比
・スライサー選択に連動するKPI

典型的な誤り: 粗利益率を計算列で行ごとに算出し、それを合計・平均する。比率は合計できないため、全体の率が実態と一致しない。比率は必ずメジャーで定義する。

基本メジャーのDAX例

① 売上金額(合計)

```
売上金額 = SUM(売上実績[売上金額])
```

② 粗利益(差分)

```
粗利益 =
SUM(売上実績[売上金額]) - SUM(売上実績[原価])
```

③ 粗利益率(比率)

```
粗利益率 =
DIVIDE([粗利益], [売上金額], 0)
```

実装上の要点: 比率は除算演算子ではなくDIVIDE関数を用い、第3引数でゼロ除算時の値を指定する。また、上位メジャーは既存メジャーを参照して定義し、同じ意味の集計式を重複させない。

事実:計算列は更新時に評価され結果を保存、メジャーは表示時に評価される | 解釈:集計を計算列で持つとフィルター条件が反映されない | 提案:集計はメジャー、分類は計算列という責務分担を規約化する

モデル設計で起きやすい問題

発生する症状は多岐にわたるが、原因の多くは「キーの重複」「関係の設定」「集計の定義場所」に集約される。原因の確認順序を固定することで、切り分けにかかる時間を大幅に短縮できる。

症状・原因・対処の対応関係

症状	主な原因	対処
合計値が想定より大きくなる	多対多、または粒度の異なるテーブルの結合による二重計上	粒度を統一し、1対多に分解する
テーブル間のキーが重複している	マスタに同一IDの行が複数存在、表記ゆれ、空白行	Power Queryで重複削除・表記統一
リレーションシップを作成できない	両側のデータ型不一致、キーの一意性欠如	型を統一し、マスタ側の一意性を確保
フィルターが別のグラフに反映されない	関係の欠落、フィルター方向の不整合、非アクティブな関係	共通ディメンション経由で絞り込む構成に変更
同じ意味のメジャーが複数存在する	担当者ごとの個別作成、命名規則の不在	基本メジャーに統合し、参照関係で構成
ExcelとPower BIで集計結果が一致しない	対象期間・除外条件・端数処理の差異、空白キー行の扱い	前提条件を明文化し、差分を項目単位で照合

切り分けの前提：症状が「特定のグラフのみ」に出るのか「モデル全体」に出るのかを最初に区別する。前者はビジュアルのフィルター設定、後者はモデル構造に原因があることが多い。単一のグラフを疑う前に、テーブル単体の合計値を確認する。

原因を確認する順番

STEP 1｜数値そのものを確認

ファクトテーブル単体の行数と合計値を、元データの値と突き合わせる。

STEP 2｜キーの一意性を確認

マスタ側IDの重複・空白の有無を確認する。重複があれば以降の調査は無効。

STEP 3｜関係の設定を確認

カーディナリティ・フィルター方向・アクティブ状態を1本ずつ点検する。

STEP 4｜集計定義を確認

メジャーの定義式と、計算列で集計していないかを確認する。

STEP 5｜ビジュアル設定を確認

グラフ単位のフィルター・相互作用設定を最後に確認する。

事実：同一の症状が複数の原因から生じる｜解釈：ビジュアル側から調査すると原因特定に時間を要する｜提案：データ→キー→関係→集計→ビジュアルの順に、下層から確認する手順を標準化する

パフォーマンスを高める設計

性能改善の起点はグラフの数ではなく、読み込む列と値の種類数（カーディナリティ）である。不要な列を読み込まず、前処理をPower Queryに寄せ、スタースキーマを維持することが最も効果的な対策となる。

6つの設計指針

① 使用しない列を読み込まない

メモリ消費は行数より列数と値の種類数に強く依存する。分析に使わない備考・自由記述列は読み込み時に除外する。

② 高カーディナリティ列を減らす

伝票番号や自由記述などの一意な文字列は圧縮が効きにくい。必要なものだけを残し、他は集約・分割する。

③ データ型を適切に設定する

数値は整数・小数、日付は日付型に統一する。数値を文字列で保持すると集計不可・容量増加を招く。

④ Power Queryで前処理を行う

列の削除・型変換・表記統一は読み込み前に実施する。可能な範囲でデータソース側に処理を委ねる。

⑤ DAXの計算列を増やしすぎない

計算列は結果を保存するため容量が増える。行単位で必要な分類のみに限定し、集計はメジャーで行う。

⑥ スタースキーマを維持する

マスタを連鎖させず、ファクトに直結する星形を保つ。関係の経路が短いほど集計処理は軽くなる。

優先順位：まず①②（読み込む列と値の種類数の削減）を実施する。ここで効果が出ない場合に④⑤⑥を検討する。ビジュアルの削減や表現の簡素化は、モデル側の改善を終えた後の手段とする。

接続モードの使い分け

Import（既定・推奨）

データをモデル内に取り込み、圧縮して保持する。

- 表示速度が最も速い
- DAXの機能制約が少ない
- データは更新時点の内容となる

DirectQuery

表示時にデータソースへ問い合わせる。

- 最新データを参照できる
- 応答はデータソース性能に依存する
- 利用できる機能に制約がある

複合モデル

テーブル単位で両方式を併用する。マスタはImport、大容量の明細はDirectQueryなど。設計・検証の負荷は高い。

事実：Importは列単位の圧縮によりモデル内にデータを保持する | 解釈：性能問題の多くは列数・カーディナリティに起因する | 提案：接続モードの変更より先に、読み込む列の見直しを実施する

実践：売上分析モデルの設計手順

設計はKPIの定義から始め、粒度の決定、テーブル分割、関係設定、メジャー作成の順に進める。この順序を守れば、検証（STEP8）で発見される不整合は限定的な範囲に収まる。

9ステップの実施手順

#	ステップ	具体的な作業	完了の判断基準
1	分析したいKPIを決める	売上・粗利・達成率など、意思決定に使う指標を列挙する	指標と定義式が文書化されている
2	データの粒度を決める	1行が表す単位（明細／伝票／日次）を選択する	粒度が1文で説明できる
3	ファクトとディメンションを分ける	数値＝ファクト、分析軸＝ディメンションに振り分ける	実績側に名称列が残っていない
4	主キーと外部キーを整理する	マスタの一意性確認、データ型の統一、空白の補完	ID重複・空白がゼロ件
5	リレーションシップを設定する	1対多・単方向で接続し、星形構造を確認する	双方向・多対多が存在しない
6	日付テーブルを作成する	連続日付を生成し、「日付テーブルとしてマーク」する	期間比較が正しく計算される
7	基本メジャーを作成する	売上金額・粗利益・粗利益率を定義し、上位指標から参照する	同義のメジャーが重複しない
8	合計値とフィルター動作を検証する	既存帳票との突合、スライサー連動と空白行の確認	基幹システムの数値と一致する
9	命名規則と説明を整備する	テーブル・列・メジャー名の統一、説明欄への定義記載	第三者が定義を確認できる

完成する分析例

月別売上推移

日付テーブルの年月を軸に、売上金額メジャーを集計

商品カテゴリ別売上

商品マスタのカテゴリ列を軸に構成比を把握

顧客別粗利益

顧客マスタを軸に、粗利益・粗利益率を並列表示

担当者別達成率

目標値テーブルを追加し、実績÷目標をメジャー化

前年同月比

マーク済み日付テーブルを前提に、タイムインテリジェンス関数で算出

事実：STEP1〜7が完了しないとSTEP8の検証は成立しない | 解釈：手順を飛ばした場合の手戻りは、粒度・キー設計に遡ることが多い | 提案：STEP2とSTEP4を関係者レビューの必須通過点とする

まとめ・設計チェックリスト

Power BIの品質はビジュアルの美しさではなく、数値の信頼性を支えるデータモデルによって決まる。公開前に以下9項目を確認し、すべてが満たされていることをレビューの通過条件とする。

公開前チェックリスト（9項目）

#	領域	確認項目	満たされない場合に発生する事象
1	構造	ファクトテーブルの粒度は明確か	合計値の二重計上、検証不能
2	構造	ディメンションのIDは一意か	リレーションシップを作成できない
3	関係	リレーションシップは原則「1対多」か	集計の対応関係が一意に定まらない
4	関係	双方向・多対多を安易に使っていないか	想定外の絞り込み、循環参照
5	時間軸	独立した日付テーブルがあり、マーク済みか	前年比・累計が正しく算出されない
6	計算	集計はメジャーで定義しているか	フィルターに反応せず数値が不一致
7	性能	不要な列を読み込んでいないか	容量増加、更新・表示の遅延
8	検証	Excelや基幹システムの数値と一致しているか	レポートへの信頼が失われる
9	運用	第三者が理解できる名称になっているか	同義メジャーの乱立、属人化

結論：Power BIの品質は、ビジュアルの美しさではなく、数値の信頼性を支えるデータモデルによって決まる。粒度・キー・関係・集計定義の4点を設計段階で固めることが、速く壊れにくいレポートへの最短経路である。

事実：9項目はいずれもモデル層の設計事項 | 解釈：ビジュアル層の作業では解消できない問題群である | 提案：公開前レビューのチェックシートとして本ページを運用に組み込む

Contact

問い合わせ

本資料の内容、データモデルの設計レビュー、Power BI導入・運用定着のご相談は、下記までお問い合わせください。

Start Challenge
Consulting co., ltd.
挑む力が、世界を照らす

データ活用・BI導入支援／業務プロセス設計

データモデル設計から運用定着までを一貫して支援します。

お問い合わせ先	
会社名	Start Challenge Consulting
メールアドレス	info@startchg.com
担当	平出

ご相談いただける内容

・既存レポートのデータモデル診断（粒度・キー・関係の点検）
・スタースキーマへの再設計、メジャーの標準化・命名規則の整備
・ExcelとPower BIの数値差異の原因調査
・社内の設計ガイドライン策定、運用定着に向けた勉強会の実施

結び：データモデルは一度作れば終わりではなく、業務の変化に合わせて見直す資産である。設計レビューの段階からご相談いただければ、手戻りを最小限に抑えられます。