

【詳細解説レポート】MaaS 事業者における経営分析システム（Sol）のデータベース設計と課題解決

1. プロジェクト背景と経営分析システム（Sol）の導入目的

現代のエンタープライズ IT インフラにおいて、日々発生するトランザクション処理に特化した基幹系データベース（OLTP：OnLine Transaction Processing）のデータを、いかにタイムリーかつ多角的に経営意思決定や戦略立案に活かすかという問題は、企業の競争力を左右する核となる。この目的を果たすシステム領域が「Sol（Systems of Insight）」であり、データ駆動型経営の要として機能する。本レポートでは、個人向けカーシェアリングサービスを展開する MaaS（Mobility as a Service）事業者である P 社を対象に、既存の貸出管理システムから経営分析システム（Sol）へのデータ活用プロジェクトの設計思想、直面した課題、および技術的解決策を詳細に論述する。

事業規模とデータ資源の構造的可視化

大都市圏を中心に急成長を遂げる P 社の事業形態、保有するデータ資源、およびシステム構築体制の全体像は以下の通りに集約される。

項目	詳細内容	設計上の意義・影響
事業形態・拠点規模	個人向けカーシェアリング（MaaS）、500 駐車場、自動車（車両）約 2,000 台、10 支社（地域運営担当）、本社統括	全国拠点からの多角的なアクセスと拠点別集計（地区別・駐車場別分析）の基盤要件
既存 OLTP 蓄積期間	貸出管理システム（関係データベース）内に過去 5 年分の貸出予約・実績データを保持	分析対象としての過去 5 年間分のプライマリデータソース
目標データ蓄積期間	表計算ソフトにて年月日別・駐車場別・車種別に保持。過去 3 年分のみ存在（3 年より前は破棄済み）	過去 5 年間の長期トレンド分析において直接的なボトルネックとなるデータ制約
プロジェクト体制	本社情報システム部（データエンジニア Q さん、データサイエンティスト R さん）、経営企画部	業務要件の抽出（R さん）、データモデル・ETL/DWH 設計（Q さん）、経営意思決定要件の策定（経営企画部）の三位一体体制

P 社本社では、サービス運営状況を多角的（地区別、性別・年代別、駐車場別、車種別）にタイムリーへ分析し、車両の追加整備計画や新サービス企画などの経営判断を迅速化するため、既存貸出管理システムをデータソースとする経営分析システムの新規導入を決定した。

Sol 領域におけるデータエンジニアの役割と技術的評価

デジタルトランスフォーメーションが加速する現代において、Sol 領域で扱うデータの量・種類・速度（Volume, Variety, Velocity）は増大の一途をたどっている。単にソースデータをそのまま移送するだけでは、複雑な集計クエリによるパフォーマンス低下やデータ整合性の不一致を招く。したがって、Sol 構築を担うデータエンジニアには以下の能力が不可欠とされる。

- **データモデリング機能:** 業務要件の集計軸（ディメンション）と数値指標（メジャー）を整理し、検索性能に特化した多次元モデル（スタースキーマ等）へと昇華させる設計能力。
- **高度なデータ操作知識:** 異種データソースからの抽出・変換・書き込み（ETL）における SQL 最適化、および一貫性と性能を両立させるバッチ処理ロジックの構築能力。経営分析システムを成功に導くためには、技術的要素の検討に先立ち、経営企画部が求める業務要件の精密な把握と、データソース側に存在する構造的制約の分析・妥協点（トレードオフ）の管理を進める必要がある。

2. 業務要件の実現可能性評価とデータ制約の分析

Sol の構築においては、経営層や業務部門が要求する理想的な業務要件に対し、既存データソースが保持する精度や保存期間の限界がギャップを生じさせることが少なくない。このギャップを早期に特定し、戦略的リスク管理のもとで現実的な実現範囲を確定することは、システム導入後のミスマッチを防止する上で極めて重大なプロセスである。

実現不可能な業務要件の特定と時系列論理メカニズム

経営企画部が策定した業務要件の評価を実施した結果、ソースデータの保持期限に起因する重大な実現不能事項が判明した。

要求と制約の不一致

- **要求業務要件:** 「過去 5 年間について、分析対象期間を柔軟に変更して、平均車両稼働率の目標比や前年同期比の傾向を分析できること」
- **ソースデータの制約:** 貸出管理システム内の予約・実績データは過去 5 年分存在するが、比較対象となる表計算ソフトの「平均車両稼働率の目標データ」は「過去 3 年分」しか蓄積されておらず、4 年前・5 年前の目標データは既に破棄されていて復元不可能である。

稼働後 2 年間の表示制限に至る蓄積ロジック

過去 5 年間の目標比分析を行うには、「実績データ（5 年分）」と「目標データ（5 年分）」の双方が揃っている必要がある。しかし、稼働初期においては目標データが 3 年分しか存在しない。システム稼働開始後に、新規発生する目標データを日次・月次でデータベースへ蓄積し続けたとしても、連続した「過去 5 年間分」の目標データが DB 上に揃うまでには、稼働開始から丸 2 年間の運用期間を経過しなければならない。したがって、**「システム稼働後 2 年間は、過去 5 年間分の平均車両稼働率の目標比を表示できない」**という明確な結論が時系列の論理プロセスとして導き出される。

制約と合意事項の構造化

経営企画部と情報システム部（Q さん・R さん）による協議の結果、業務要件の変更・退行を避けつつプロジェクトを進行させるため、以下の構造化された対応方針が採択・合意された。

- **要求要件（将来像の確定）:**
- 過去 5 年間を対象とした分析対象期間の柔軟な変更。

- 年別・月別・日別・週別・曜日別などの時間軸における平均車両稼働率の目標比および前年同期比分析。
- **データソース制約（物理的限界）：**
- 平均車両稼働率の目標データは過去 3 年分のみ保持（3 年より前は破棄済み）。
- **合意された対応方針：**
- 業務要件の将来的な定義そのものは変更しない。
- システム運用開始直後の 2 年間は、データが存在する「実現可能な範囲（過去 3 年分から順次拡大）」で目標比分析を実施・運用する。判明したデータ制約を前提とし、次に問合せ処理性能を最適化し多角的な集計を可能とする多次元データモデルの策定へと進む。

3. 多次元データベースのモデル設計（スタースキーマ構造）

経営分析システムにおける多次元データベース設計では、複雑な検索・集計クエリに対するレスポンス性能の向上と、多次元の集計軸（時間、地区、車種、会員属性など）による自由度の高いスライシング／ダイシング操作の両立が絶対条件となる。

OLTP と Sol のモデル対比およびスタースキーマの採用理由

本システムでは、行指向データベースを用いた多次元データベース構造として**「スタースキーマ構造」**を採用した。

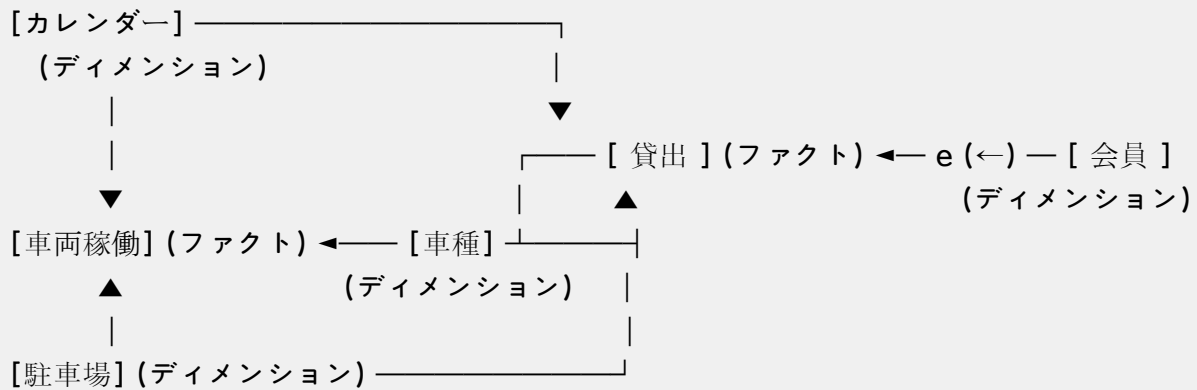
比較観点	OLTP（貸出管理システム：図 1）	Sol（経営分析システム：図 3）
設計思想	第 3 正規形（正規化）	非正規化（スタースキーマ）
最適化対象	高頻度な更新・挿入処理時の更新異常防止と整合性保持	大規模データの参照・多角的なクロス集計クエリの超高速化
構造の特徴	テーブル分割による重複の徹底排除（多数の表が存在）	中心のファクト表を囲むようにディメンション表を配置
JOIN 処理コスト	多数の表を頻繁に結合するため、集計クエリ実行時の負荷が高い	ファクト表と各ディメンション表の 1 段階の結合に抑えられ、性能が向上

行指向データベース環境において、正規化された OLTP 構造のまま多期間・多軸の集計クエリを発行すると、表結合（JOIN）の階層が深くなり走査コストが肥大化する。Sol 側でスタースキーマを採用することは、JOIN 処理の簡素化と構造の単純化をもたらし、問合せ性能を最大化させるアーキテクチャ的必然性を持つ。

エンティティ間関連および属性補完の精密解析

貸出管理システムのデータ構造（図 1）から経営分析システムのデータモデル（図 3）への変換にあたり、各エンティティの参照関係および追加属性の論理根拠は以下の通り規定される。

【図 3 に準拠したスタースキーマ構造（経営分析システム）】



決定根拠と属性補完の解説

- **b 年月日**（「車両稼働」ファクト表の主キー属性）：「車両稼働」ファクト表は、「カレンダー」ディメンション表と多対 1 の参照関係を持つ。「カレンダー」側の主キーである「年月日」を参照して日別の傾向分析を行うため、「車両稼働」の主キーの一部（兼 外部キー）として「年月日」を設定する。
- **c 年代 / d 貸出実績件数**（「貸出」ファクト表の属性）：業務要件である「会員の性別・年代別の人気車種分析」に対応するため、会員属性から導出される「年代」（属性 c）を保持させる。また、「人気車種の分析（貸出実績件数が多い車種を人気車種とみなす）」に対応するため、集計値である「貸出実績件数」（属性 d）をファクト表のメジャー（指標）として追加定義する。
- **e ←**（参照関係：1 対多関連）：「会員」ディメンション表（1）と「貸出」ファクト表（多）の関係を示す。1 人の会員は複数の貸出を行うため、関連の向きは貸出 ← 会員（主キー参照：貸出側が会員 ID を外部キーとして保持）となり、記号としては ← を設定する。設計されたデータモデルに対し、正確な集計データを継続的に供給するためのバッチ処理プログラムおよび SQL 構文の精緻な評価を行う。

4. データ加工処理（ETL）と SQL 構造の検証

基幹系 OLTP（図 1）から分析系 Sol（図 3）へデータを移送・加工するバッチ処理では、異なるテーブル構成から論理条件を抽出し、データ整合性を維持した上でファクト表へ格納する精度が求められる。

SQL 構文のステップバイステップ解析

「貸出」ファクト表に格納する「遅延返却発生件数」を集計・作成するために開発されたバッチ処理用 SQL 文（図 4）の構文構造とキー条件の解析は以下の通りである。

```

SELECT R.貸出予定年月日, R.駐車場 ID, R.車種 ID, R.会員 ID, COUNT(*) AS 遅延返却発生件数
FROM (SELECT Y.貸出予約コード, Y.駐車場 ID, Y.車種 ID, Y.会員 ID,
      TIMESTAMP_TO_DATE(Y.貸出予定時刻) AS 貸出予定年月日, Y.返却予定時刻 FROM
      貸出予約 Y) R
  
```

```
INNER JOIN 貸出実績 J ON R.貸出予約コード = J.貸出予約コード
WHERE R.返却予定時刻 < J.返却実績時刻
GROUP BY R.貸出予定年月日, R.駐車場 ID, R.車種 ID, R.会員 ID
```

(※一次情報源の表記に基づき、列名内のスペースを含めて記述している)

空欄 f および g の技術的導出理由

- **f INNER JOIN 貸出実績 J ON R.貸出予約コード = J.貸出予約コード**
- **論理構造:** 遅延返却（返却予定時刻 < 返却実績時刻）を判定するためには、予約データ（副問合せ結果 R）と実際の返却時刻を保持する「貸出実績」（J）の突き合わせが必須である。図 1 において両者は 貸出予約コード を共通のキー（1 対 1 の関連）として結びついているため、内部結合（INNER JOIN ... ON ...）を適用する。
- **技術注記:** 標準 SQL において INNER はデフォルトの結合方式であるため、実務上 JOIN 単体でも同等に動作するが、結合の意図を明確にする表記として INNER JOIN が設定される。
- **g GROUP BY R.貸出予定年月日, R.駐車場 ID, R.車種 ID, R.会員 ID**
- **論理構造:** SELECT 句で集計関数 COUNT(*) を適用して遅延件数を算出しているため、GROUP BY 句には SELECT 句で指定した集計対象以外のすべての非集約列（貸出予定年月日, 駐車場 ID, 車種 ID, 会員 ID）を網羅してグループ化の単位を明示する必要がある。

バッチ処理の実行頻度とゼロパディングによるデータ品質管理

本バッチ処理を実行すべき頻度は**「日次」**である。

- **導出根拠:** 業務要件（図 2）に「遅延返却発生件数については前日までの実績を翌営業日の朝に確認できること」と明記されている点、および「貸出」表の主キーが「年月日」単位で構成されていることから、夜間バッチによる「日次」自動実行が導き出される。
- **データクレンジング・ゼロ補完（ゼロパディング）の意義:** SQL の COUNT(*) による集計のみでは、ある日に特定駐車場・車種・会員の組み合わせで遅延が 1 件も発生しなかった場合、その組み合わせのレコード自体が抽出されない。バッチプログラム側で「遅延が発生しなかった組み合わせに対して遅延返却発生件数=0 件のレコードを明示的に挿入（ゼロパディング）」するロジックを組むことで、後続の BI ツールやクエリ発行時に多重の外側結合（OUTER JOIN）を行わずに済み、データの抜け落ち防止と検索処理の安定化が保証される。日次バッチによってデータが定期的に蓄積されると、数年後にはファクト表のデータ量が膨大となり、レスポンス性能の悪化という新たな課題が生じる。

5. 分析レスポンス性能の改善策（データマートの導入）

蓄積データが過去複数年間に及ぶと、行指向データベース上に適切なインデックスを配置していても、集計クエリが走査すべきデータブロック数の増大によりレスポンス低下が避けられなくなる。経営意思決定のスピードを殺さないためには、事前集計表（データマート）の導入による集計コストの削減が極めて有効である。

ボトルネックの特定と改善アプローチ

経営企画部から「過去複数年間、時間軸を月別として人気車種及び遅延返却発生件数を分析する際、処理に時間がかかりすぎる」という性能課題が提起された。「貸出」ファクト表のデータ粒度は「日次（年月日）」である。そのため、「過去複数年間の月別トレンド」を抽出するクエリを発行するたびに、数百万～数千万行に及ぶ日次ファクト表の該当範囲を走査（インデックスレンジスキャンまたはフルスキャン）し、実行時に月単位へ再集約する計算コストが発生していた。これがボトルネックの正体である。この対策として、データエンジニア Qさんは「月単位」で事前集計を済ませた集計表（データマート）をデータベースに追加定義した。

データマートの構造設計と業務要件への直接的マッピング

追加された月次集計表（データマート）と既存の日次ファクト表のキー構造および粒度の比較は以下の通りである。

比較要素	日次ファクト表（「貸出」表）	追加された集計表（データマート）
時間軸の粒度	日次（年月日）	月次（年月）
集計演算コスト	クエリ発行時に日次データを都度集計（高負荷）	夜間バッチで事前に月単位集計済み（極めて低負荷）
主キー構成	年月日, 駐車場 ID, 車種 ID, 会員 ID	年月, 駐車場 ID, 車種 ID, 会員 ID

主キー構成（設問4 解答）と業務要件の直接的対応関係

追加された集計表の主キーが **年月, 駐車場 ID, 車種 ID, 会員 ID** である理由は、経営企画部の要求要件と一対一で対応している。

- **年月**：業務要件「時間軸を月別として」の集計単位に対応。
- **車種 ID**：業務要件「人気車種（貸出実績件数が多い車種）」の集計単位に対応。
- **駐車場 ID, 会員 ID**：業務要件「駐車場別・会員別の遅延返却発生件数」の集計単位に対応。この主キー構成で事前集計表を保持することにより、過去複数年にわたる大量データに対するクロス集計であっても、走査対象行数を劇的に削減し、秒単位でのレスポンスを実現することが可能となった。

6. 結論と MaaS データベース設計における知見の要約

本事例の解析を通じて、**Sol** 領域における経営分析システムの構築では、単に静的なデータモデルを構築するにとどまらず、業務制約のトレードオフ管理から **ETL** 処理の完全性確保、さらには将来のデータ肥大化を見据えた性能対策に至るまで、一貫した技術アプローチが不可欠であることが実証された。

コア・アウトカムの要約

1. **業務要件とデータ制約のトレードオフ管理** データソース側の限界（目標データ 3 年保持上限）を即座に割り出し、業務要件の定義を崩すことなく**「稼働直後の 2 年間は過去 5 年分の目標比表示を行わない」**という現実的かつ妥当な運用範囲を経営企画部と合意形成した。

2. **スタースキーマによる多次元検索の最適化**行指向データベースにおける検索・集計性能を最大化させるため、**スタースキーマ構造**を採用し、分析軸となる属性（年代、貸出実績件数等）および1対多関連（←）を正確にモデリングした。
3. **日次 ETL 処理の自動化とゼロ補完による品質維持** INNER JOIN および GROUP BY を組み込んだ**日次バッチ処理**を確立し、遅延0件レコードの補完挿入（ゼロパディング）を実施することで、多次元分析クエリの整合性と精度を保全した。
4. **月次データマート導入によるレスポンス保障**日次粒度データに対する複数年集計のボトルネックに対し、業務要件に直結する **年月, 駐車場 ID, 車種 ID, 会員 ID** を主キーとする**月次集計表**を追加し、問合せパフォーマンスを抜本的に改善した。MaaS事業をはじめとする高度なデータ活用を目指す企業において、Sol の真価を引き出す鍵は、ビジネス側の分析要件、論理データモデリング、および物理データベース性能対策の3要素を高度に整合させるデータエンジニアリングの実践に他ならない。