DM2.0
クエリ言語仕様書

Version 1.02

ダイナミックマップ 2.0 コンソーシアム

名古屋大学大学院情報学研究科付属 組込みシステム研究センター
名古屋大学 COI 人がつながる　“移動“イノベーション拠点

2017/09/06

更新履歴

Version	日付	内容	担当
1.00	2017/03/24	新規作成	渡辺
1.01	2017/08/29	仕様公開のために用語の基本説明を追加 道路地図仕様の呼称を「名古屋大学 COI 高精度地図フォーマット」に統一	渡辺
1.02	2017/09/05	誤字修正	渡辺

目次

1. はじめに 2

1.1. 本書の概要及び位置づけ 2

1.2. 前提知識 2

2. ダイナミックマップにおけるクエリ言語の意義 3

2.1. ダイナミックマップとは？ 3

2.2. プログラミング言語による情報の検索・統合の問題点 4

2.3. なぜクエリ言語を使うのか？ 4

3. クエリ言語の要件 5

4. クエリ言語 7

4.1. 共通データモデル 7

4.1.1. 動的（準動的）情報 7

4.1.2. 静的（準静的）情報 7

4.1.3. 予測情報 8

4.1.4. 地図情報 8

4.2. クエリ言語の設計指針 9

4.3. クエリ言語の文法 9

4.3.1. One-shot query のための構文規則 10

4.3.2. Continuous query のための構文規則 10

4.3.3. 再帰検索のための構文規則 11

5. 記述例 13

5.1. 動的情報単体への検索 13

5.2. 道路地図の検索(1) 13

5.3. 道路地図の検索(2) 14

5.4. 道路地図の検索(3) 15

5.5. 動的情報と道路地図の統合 16

5.6. 道路地図を介した動的情報同士の統合 17

終わりに 19

謝辞 19

図 一覧

図 1　ダイナミックマップとクエリ言語 4
図 2　動的情報のリレーション表現 7
図 3　静的情報のリレーション表現 8
図 4　予測情報のリレーション表現 8
図 5　クエリ言語における系譜 9
図 6　ONE-SHOT QUERY のための構文規則 10
図 7　CONTINUOUS QUERY のための構文規則 11
図 8　再帰検索のための構文規則 12
図 9　動的情報単体の検索（車両 ID のフィルタリング） 13
図 10　指定した点を包含するレーンの検索 14
図 11　経路接続の関係を使った次レーンの検索 15
図 12　レーンに対する再帰検索 16
図 13　車両の位置座標を包含するレーン領域の検索 17
図 14　地図を介した動的情報同士の統合 18

1. はじめに

本書はダイナミックマップ 2.0 コンソーシアムにて構築する、データベースシステムのクエリ言語に関する仕様書である。

1.1. 本書の概要及び位置づけ

本書は、クエリ言語に関する 2016 年度の検討結果をまとめ、翌年度のクエリ言語処理系（クエリパーサ、コンパイラ等）の設計、実装の前提となる機能、文法を記述することを目的とする。また、クエリの記述例なども記載することで、DM2.0 システムを用いたデータ検索のイメージを伝える。

1.2. 前提知識

本書はリレーショナルモデルとリレーショナル代数演算、そして SQL に関する初歩的な知識を有していることを前提に記述されている。そのため、普段データベースと関わりの少ない分野の読者には読みづらくなってしまっている点は認める。リレーショナルデータベース技術の技術書や教科書[(1),(2)]を片手に参照されることを薦める。

また、クエリの記述例に出てくる地図情報は名古屋大学 COI 高精度地図フォーマットをベースにしているが、本書には名古屋大学 COI 高精度地図フォーマットの仕様に関する説明が含まれていない。名古屋大学 COI 高精度地図フォーマットの仕様書は本書と同時に公開されているので、そちらは仕様書を片手に参照してもらうことを期待している。

(1) 増永良文, 「リレーショナルデータベース入門 – データモデル・SQL・管理システム -」, サイエンス社.

(2) 北川博之, 「データベースシステム」, オーム社.

2. ダイナミックマップにおけるクエリ言語の意義

本節ではダイナミックマップのためのクエリ言語の必要性について説明する。

2.1. ダイナミックマップとは？

近年の交通分野では、車両に搭載されたセンサにより走行環境を認識し、ドライバへの警告や自動で危険を回避する高度な安全運転支援システム、自律走行を可能とする自動走行システムの研究・開発が加速している。また、交通流の円滑化、環境負荷軽減などを目的として、車両と道路インフラが連携する、協調 ITS(Intelligent Transport Systems)の利用も活発化している。しかし、単体のセンサで認識できる範囲は非常に限定的（レーザレーダの場合、射程は約 120m）であり、手前の物体に遮られて奥の物体が検知できない等の問題もある。複数の車両や道路インフラの間で、道路上の事象を検出したセンサ情報を共有できるようにすることが、交通サービスを発展させていく上でますます重要となっている。

車載システムの進化に伴い、道路地図の高度化も必要となっている。これまでの車載システムでは、道路地図を目的地までの経路を案内するナビゲーションに主に用いてきた。それに加えて今後は、自動運転システムが自車両の位置を推定するための情報として、周辺の地形・建物の凹凸などの空間特徴量を提供することや、車両や道路インフラから得られたセンサ情報に対して、交通ルール上の意味付け（どのレーン上の事象なのか、そのレーンと自分のいるレーンとはどういう関係なのか等）を提供することなど、新たな役割が道路地図に期待されるようになっている。そのようなニーズに応えるための高精度道路地図の作成に向けて、国内外の様々な組織で仕様検討や試作が行われている。

高精度の道路地図上に、センサなどから得た交通データ（動的情報，準動的情報，準静的情報）を重ねて、位置参照方式を用いてお互いに紐づけられるようにしたデータ集合は「**ダイナミックマップ**」と呼ばれている。ヨーロッパで提案された当初は、車両周辺の情報を扱うローカルダイナミックマップとして検討されてきたが、現在は広域を扱うように概念が拡張されており、ダイナミックマップは、自動走行システム等の高度な交通サービスを支えるために必要な情報基盤と位置づけられている。以下はダイナミックマップを構成する各種情報の代表例である。

- 静的情報：道路地図
- 準静的情報：標識、ランドマーク、路側設備など
- 準動的情報：渋滞、道路工事、路面状態など
- 動的情報：移動体（車両、人など）の位置情報、信号の情報など

2.2. プログラミング言語による情報の検索・統合の問題点

ダイナミックマップは、交通分野において提供される様々な種類の情報により構成されるデータ集合である。それらの情報は単独で検索に利用されるだけではなく、地図などを介して互いに重畳・統合されることによって、単一種類の情報からでは得られない知識の抽出を可能とする。多種多様な情報に対する検索・統合をどのように扱うのかという問題が、ダイナミックマップにおいては非常に重要である。

現在の交通分野のアプリケーションプログラム開発、とくに車載組込み開発の分野では、データ管理を DB システム等のミドルウェアに頼らずにプログラミング言語単独で実現する場合が多い。プログラミング言語だけでも情報の検索・統合は実現可能ではあるが、そのためには処理手順すべてをコードに落とし込む必要がある。それはすなわち、コード記述量が多くなるということであり、プログラマの負担増加や開発期間の長期化を意味する。ダイナミックマップが扱う情報の多様さとその統合の複雑さが上がっていくほど、プログラミング言語だけの実現ではアプリケーションプログラム開発のコストが高くなる懸念がある。

2.3. なぜクエリ言語を使うのか？

上記に対する対策として、ダイナミックマップ 2.0 コンソーシアムでは情報に対する検索・統合のためのクエリ言語を導入する。この考え方は、リレーショナルデータベース管理システム(RDBMS: Relational Database Management System)におけるクエリ言語のコンセプトと共通である。

情報の検索・統合において、(1)「**どういうデータを探してほしいのか**」と(2)「**どういう手順でデータを探してほしいのか**」は分離することが可能である。プログラマにはクエリ言語を使って前者(1)についてのみ指定させる。クエリ内では、検索・統合の結果となる各データが満たすべき条件や、データの構造のみを宣言する。後者(2)については、クエリの内容を満たす情報の検索手順自体の導出や、検索手順の効率化・高信頼化等をシステム側へ一任する。プログラマを(2)から解放することで、開発コストの削減を狙う。

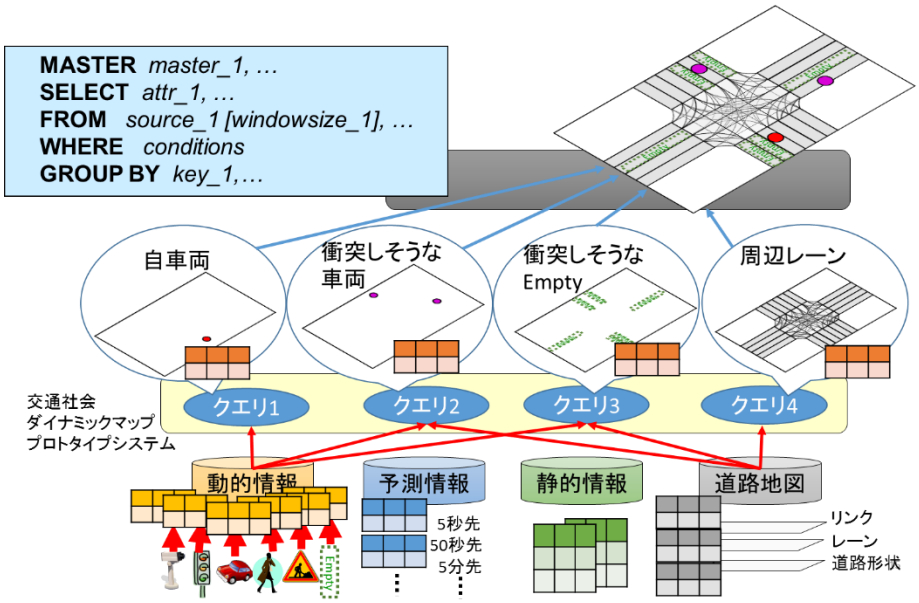


図 1　ダイナミックマップとクエリ言語

3. クエリ言語の要件

ダイナミックマップのクエリ言語が満たすべき要件について述べる。

- **共通データモデルを扱えること**：
  ダイナミックマップが扱うことになる、動的(準動的)情報、静的(準静的)情報、予測情報、地図情報について、それぞれを個別のデータモデルで表現してしまうと、情報を統合する際にデータモデル間の違いが障壁となる。ダイナミックマップ全体で、どの種類の情報も表現できる共通のデータモデルを導入するべきで、クエリ言語はそれを扱うことができなければならない。

- **One-shot query を扱えること**：
  既に蓄積された状態のデータの集合に対して、クエリ内の条件を満たすものだけを 1 度だけ取得するというクエリの処理形態を **One-shot query** と呼称する。One-shot query は RDBMS などに蓄積されたデータに対するクエリ処理形態である。クエリ言語は One-shot query の指定ができなければならない。

- **Continuous query を扱えること**：
  先にクエリを登録し、データの生成や時間の経過に連動してクエリの条件を満たすかどうかの評価を繰返し実行するクエリの処理形態を **Continuous query** と呼称する。Continuous Query はストリームデータに対するクエリの処理形態である。クエリ言語は Continuous Query の指定ができなければならない。

- **情報統合のための機能を備えること**：
  ダイナミックマップでは、各種情報を互いに重畳・統合することが定常的に行われる。クエリ言語は情報統合のための機能を提供しなければならない。具体的にはリレーショナル代数演算における結合(Join)や、データフュージョン、センサーフュージョンなどの機能である。

- **実行タイミング指定ができること**：
  Continuous query によって条件の評価を繰り返し実行する際に、実行タイミングの指定がクエリ言語内でできなければならない。具体的には、データの到着時や、時間の経過時などがタイミングとして指定できることが望ましい。

- **移動ウインドウを扱えること**：
  ストリームに対して Continuous query の処理を実施する場合、逐次到着するデータをいつまで蓄えておくかを指定できなければならない。データを保持する条件として、時間幅やデータ件数などが扱えることが望ましい。データを保持する条件は、時間経過やデータ到着に合わせて移動することから、この機能は**移動ウインドウ**と呼称される。移動ウインドウの有効範囲外となったデータは、システムによって削除される。

- **グラフの再帰検索のための機能を備えること**：
  ダイナミックマップに含まれる地図情報のうち、道路ネットワークや車線ネットワークは論理的にはグラフ構造をしている。グラフに対する典型的な検索要求の一つとして、任意の数の辺を経由した先にある特定の頂点を探すといったものがある。このような任意回の繰返しを含む検索を、**再帰検索**と呼称する。クエリ言語は再帰検索の指定ができなければならない。

- **ユーザ定義関数などの拡張機能を備えること**：
  将来的にダイナミックマップに含まれる可能性のある情報の種類や、それらの情報同士の統合方法を、事前に全て網羅して提供することは容易ではない。事前に網羅する代わりに、用途に合わせた機能拡張のための手段を利用者へ提供するべきである。ユーザ定義関数は機能拡張の手段の一つであり、クエリ言語はユーザ定義関数をサポートするべきである。

4. クエリ言語

本章ではクエリ言語の仕様について述べる。

4.1. 共通データモデル

ダイナミックマップ 2.0 コンソーシアムでは、ダイナミックマップ全体の共通データモデルとして**リレーション**を扱う。リレーションは、RDBMS などを中心として採用されてきたデータモデルである。動的（準動的）情報、静的（準静的）情報、予測情報、地図情報の各情報をリレーションで表現する。図による視覚化の都合から、以下ではリレーションを表構造で表すものとする。リレーション内の要素である**タプル**は、その表に属する各行として表し、あるタプルのもつ各属性の値はその行の各列として表す。

4.1.1. 動的（準動的）情報

動的（準動的）情報は、データの生成時刻を表すタイムスタンプを格納した、特殊な属性を持つリレーションとして扱う。１つのタプルはある時刻においてセンサなどから得られたデータの 1 インスタンスを表すものとする。また、一度生成されたタプルは後から別の値を上書きされない。新たなデータが得られた場合は、別のタイムスタンプがついた別のタプルとして、挿入される。このような単調増加していくデータの系列を、ストリーム（リレーショナルストリーム）と呼称する。

TimeStamp	A1	A2	⋯
⋮	⋮	⋮	⋮
12:03:00	24	56	⋯
12:03:01	23	56	⋯
12:03:02	23	56	⋯
12:03:03	23	57	⋯

図 2　動的情報のリレーション表現

例えば、図におけるタプル（”12:03:00”, 24, 56, …）は、時刻”12:03:00”において得られたセンサの値の組を表しており、その時刻における属性 A1 の値は 24, 属性 A2 の値は 56 であったことを意味している。

4.1.2. 静的（準静的）情報

静的（準静的）情報は、蓄積されたリレーションとして扱う。更新操作は可能だが、更新前の情報も後から参照できることを保障する必要があるため、値の上書きではなく、別のタプルとして挿入するものとする。各タプルにはデータの有効期間（ [開始時刻、終了時刻) ）がついており、後から特定時刻のリレーションの状態を再現できるようになっているものとする。終了時刻が定義されていないタプルは最新の情報であるとみなす。新しいタプルが挿入された場合に古いタプルの有効期間の終了時刻がセットされる。

StartTS	EndTS	ID	・・・
⋮	⋮ ⋮	⋮	
11:00:00	12:00:00	1	…
12:00:00	-	1	…
11:00:00	-	2	…
11:00:00	-	3	…

図 3　静的情報のリレーション表現

例えば、図におけるタプル(“11:00:00”, “12:00:00”, 1, …)は、時刻”11:00:00”に生成され、”12:00:00”に別のデータで上書きされた情報であることを表している。

4.1.3. 予測情報

予測情報は、ストリームの特殊なケースとして扱う。各タプルには 2 種類のタイムスタンプが付与されており、予測の元になったデータの最新時刻と、いつの状態を予測した情報なのかを表す時刻を持つものとする。

BaseTS	ForecastTS	A1	・・・
⋮	⋮ ⋮	⋮	
12:03:00	12:04:00	24	…
12:03:01	12:04:01	24	…
12:03:02	12:04:02	24	…
12:03:03	12:04:03	25	…

図 4　予測情報のリレーション表現

例えば、図のタプル(“12:03:00”, “12:04:00”, 24, …)は、時刻”12:03:00”時点の情報を元に、時刻”12:04:00”の状態を予測した情報であることを表している。

4.1.4. 地図情報

地図情報は静的情報の 1 種であるため、他の静的情報と同様に、有効期間の属性情報を付与したリレーションとして扱うことを推奨する。ただしリレーション構造の詳細は、各道路地図仕様に準拠しなければならないので、そちらを優先するものとする。道路ネットワークや車線ネットワークは、論理的にはグラフ構造であるが、リレーションに分解された状態で格納する。

ダイナミックマップ 2.0 コンソーシアムで扱う道路地図は、名古屋大学 COI 高精度地図フォーマットである。詳しくは名古屋大学 COI 高精度地図フォーマットの仕様書を参照すること。

4.2. クエリ言語の設計指針

One-shot query を扱う RDBMS (Relational DataBase Management System)の研究分野では、リレーションに対するクエリ言語は様々なものがこれまで検討されてきた。もっともよく利用されているものは、国際標準である SQL である。最初の規格は 1986 年に ANSI によって策定され、1987 年に ISO に批准された(SQL 86)。1992 年には大幅改訂された、SQL 92 が策定されている。さらに、1999 年には再帰検索やトリガ等をサポートした SQL 99 が策定された。国際標準以外の RDBMS 向けのクエリ言語では、述語論理をベースにした DataLog があり、SQL 99 同様に再帰検索ができることが知られている。

一方、2000 年前後からストリームを対象に Continuous query を実現するための、DSMS (Data Stream Management System)の研究が活発になった。DSMS の研究者は、RDBMS のクエリ言語を参考に DSMS 向けのクエリ言語を作成した。Stanford 大が提案した CQL、StreamSpinner プロジェクトの SpinQL などがある。ただしこれらは、SQL 92 をベースとしており、SQL 99 や DataLog に備わっている再帰検索の機能は含まれていない。

ダイナミックマップのクエリ言語が備えるべき要件の一つに、道路ネットワークや車線ネットワークなどのグラフに対する再帰検索がある。そのため、既存の DSMS のクエリ言語のままでは機能が十分ではない。ダイナミックマップ 2.0 コンソーシアムでは、ダイナミックマップのためのクエリ言語内での再帰検索の構文をどのようにするかについて議論を行った。SQL 99 をベースにする案と、DataLog をベースにする案を出して比較検討した結果、国際標準である SQL 99 をベースにする方が普及に有利となるだろうとの結論を得ている。

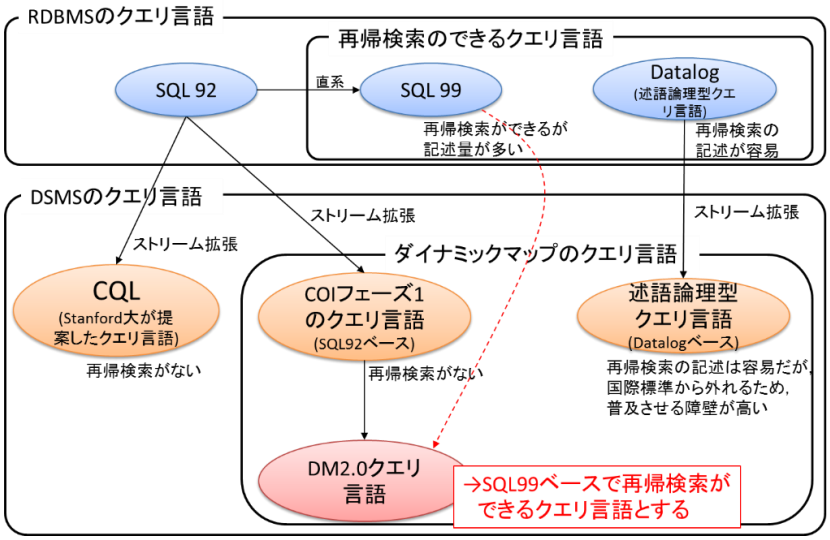


図 5　クエリ言語における系譜

4.3. クエリ言語の文法

本節ではダイナミックマップのためのクエリ言語の文法について説明する。前述したとおり、ダイナミックマップ内ではすべての情報が共通データモデルであるリレーションとして扱われている。プログラマは、クエリ言語を

使って、単体のテーブルへの検索や、複数のテーブルにまたがった統合を記述する。クエリの文法は 3 種類の構文で構成される。

4.3.1. One-shot query のための構文規則

One-shot query は、既に蓄積された状態のリレーションに対して、クエリ内の条件を満たすものだけを 1 度だけ取得する処理形態である。One-shot query を指定するための構文を図に示す。

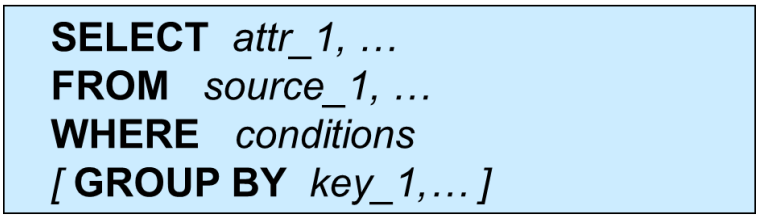


図 6　One-shot query のための構文規則

SELECT、 FROM、 WHERE、 GROUP BY 節は、SQL で用いられているものと同様の意味を持つ。SQL 自体の詳しい解説は省略する。

- FROM 節には、クエリの入力となるリレーションの名前を列挙する。複数のリレーション名が与えられた場合は、直積(Cartesian Product)や結合(Join)の処理が内部的に行われることを意味する。各リレーション名には AS を用いて一時的な別名をつけることができ、この機能は同じリレーション同士を Join する場合に必要となる。

- WHERE 節には、結果として上がってくるタプルが満たすべき条件を指定する。条件は真偽値を返す比較演算子(=、 <、 >、 >=、 <=、 <>など)を、AND や OR （現段階では NOT は含めない）などの論理演算子でつなげたものである。比較演算子以外にも、真偽値を返すものであれば関数も使用可能とする。

- GROUP BY 節は、タプル集合をグループ化して集約演算(平均、カウント、最大、最小など)を適用するための機能である。タプルをグループ化するためのキーになる属性を列挙する。GROUP BY 節の記述は必須ではなく、省略された場合には集約演算の実行は行われない。

- SELECT 節は、クエリの結果として残すべきデータの構造を指定している。FROM 節に出現したリレーションに含まれる属性名または、関数を指定可能である。AS を用いた別名の付与もできるものとする。

4.3.2. Continuous query のための構文規則

Continuous query は、先にクエリを登録し、データの生成や時間の経過に連動してクエリの条件を満たすかどうかの評価を繰返し実行するものである。Continuous query を指定するための構文を図に示す。

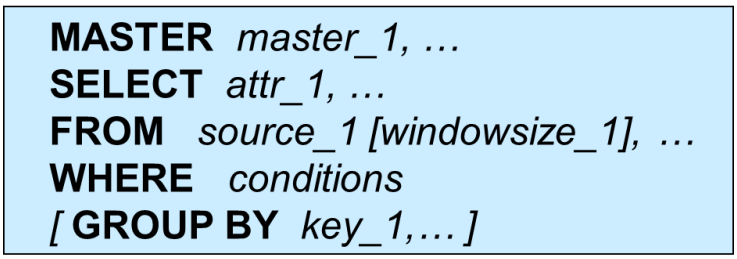


図 7　Continuous query のための構文規則

SELECT、 FROM、 WHERE、 GROUP BY 節は、one-shot query と共通だが、SQL にはないものとして新たに MASTER 節が追加されている。

- MASTER 節は、クエリを実行するきっかけを与えるイベントの指定を行うためのものである。イベントはリレーション名であり、そのリレーションの新規のデータが発生する度にクエリの評価が繰返し行われる。

- FROM 節では、リレーション名の後に移動ウインドウの指定が可能になっている。時間幅に基づく移動ウインドウと、データ件数に基づく移動ウインドウの 2 種類を指定可能とする。 移動ウインドウを省略した場合は、無限の幅を持つウインドウと解釈されるが、長期にわたって実行されるクエリでは、そのような使い方は現実的ではない。

MASTER 節に記述されたリレーションに属する新規タプルが発生すると、その度にクエリの評価イベントが発生する。その時刻の状態で FROM 節の各リレーションの移動ウインドウが一時固定され、移動ウインドウ内の各タプルに対して、SELECT、 FROM、 WHERE、 GROUP BY 節の評価が行われる。

4.3.3. 再帰検索のための構文規則

道路ネットワークや車線ネットワークなどのグラフに対する、再帰検索のクエリを指定するための構文を図に示す。One-shot query 版と Continuous query 版がある。

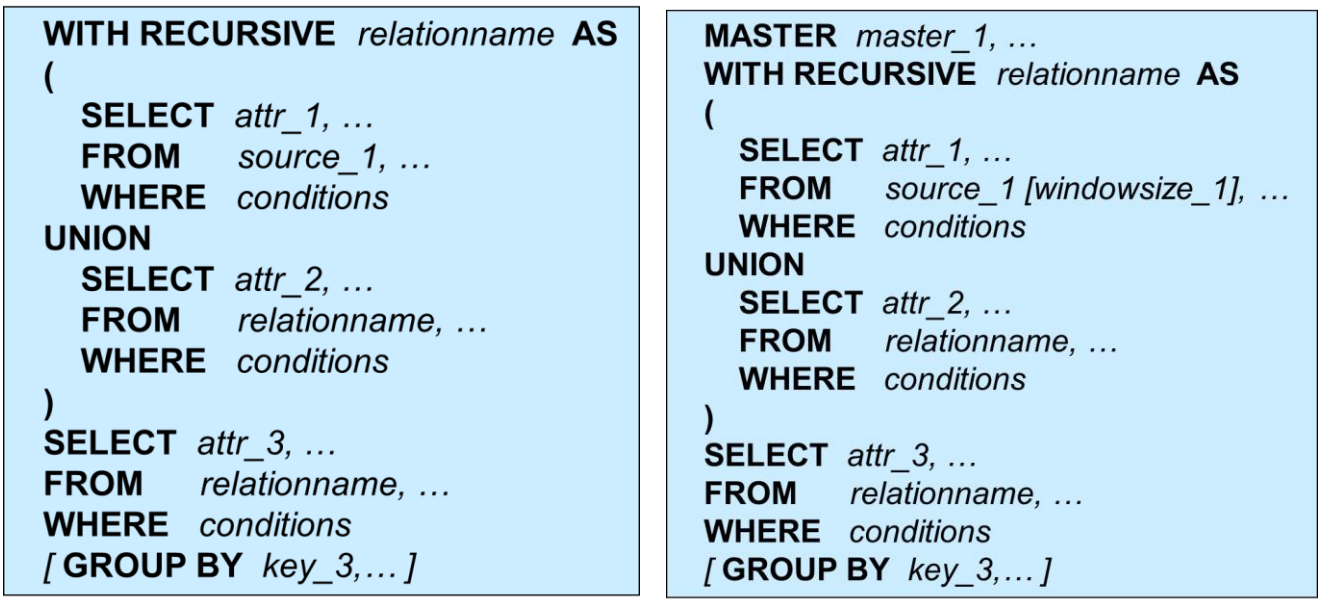


図 8　再帰検索のための構文規則

- WITH RECURSIVE 内は、カッコ内のサブクエリによって新規のタプルが生成されなくなるまで繰返し評価が実行される。繰返し処理によって生成された処理結果は一時的なリレーション(relationname で指定)へ格納される。WITH RECURSIVE は SQL99 で導入された構文をベースにしたものである。

- WITH RECURSIVE 内の記述は SQL99 では自由度が高いが、典型的な再帰検索の記述に用いる場合は以下のようにする
  - UNION の前の SELECT-FROM-WHERE には、再帰検索の初期集合を得るためのサブクエリを与える
  - UNION の後の SELECT-FROM-WHERE には、初期集合を拡張するためのサブクエリを与える。WHERE には再帰検索の終了条件が付いていることが望ましい。

5. 記述例

交通アプリケーションプログラムが必要とする、すべての局面をカバーする全クエリセットを紹介することはできないが、動的情報と道路地図情報に対する典型的なクエリの例を具体的に紹介する。ここでは道路地図情報は名古屋大学 COI 高精度地図フォーマットに基づいたレーン（車線）レベルの詳細度のものを想定している。

5.1. 動的情報単体への検索

図は「車両の動的情報をフィルタリングして、車両 ID 属性が 0 のものの位置情報だけ取得してほしい」という場合の continuous query の記述例である。

MASTER 節と FROM 節には車両の動的情報の vehicle_stream が指定されており、新規の車両の情報が到着する度にこのクエリが実行されることを意味している。WHERE 節では届いた情報の id 属性の値をチェックしており、条件を満たす場合はタプルが通過する。また、SELECT 節の指定により、到着時刻 timestamp と id と位置 position が 3 つ組で送信される。

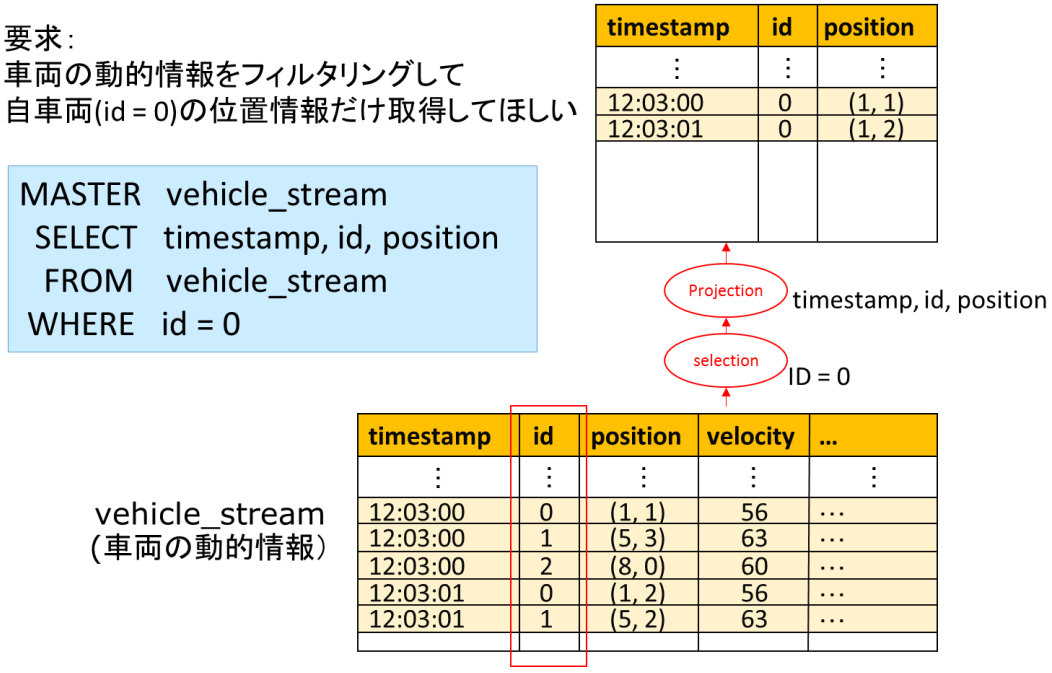


図 9　動的情報単体の検索（車両 ID のフィルタリング）

5.2. 道路地図の検索(1)

図は「特定の座標(例では座標(1，2))を包含しているレーンの走行可能領域（レーン形状）と id を知りたい」という場合の記述例である。

このクエリでは MASTER 節が含まれないため、one-shot query と解釈され、登録された際に一度だけ実行される。FROM 節では 3 種類のテーブルが記述されている。pf_area_feature は名古屋大学 COI 高精度地図フォーマットにおける面カテゴリのデータを格納するテーブル、st_face は、面の形状データを格納するためのテーブル、pf_area_topo_prim はその二つの間の関係を保持するためのテーブルである。ST_Contains は領域と点の包含関

係を判定する関数、ST_MakePoint は点データを生成するための関数で、どちらもユーザ定義の関数である。なお、p.feature_class_code には本来ならレーン領域クラスを表す 4 桁のコードが入るが、文書の読みやすさのため「'Lane Area'」と表記した。

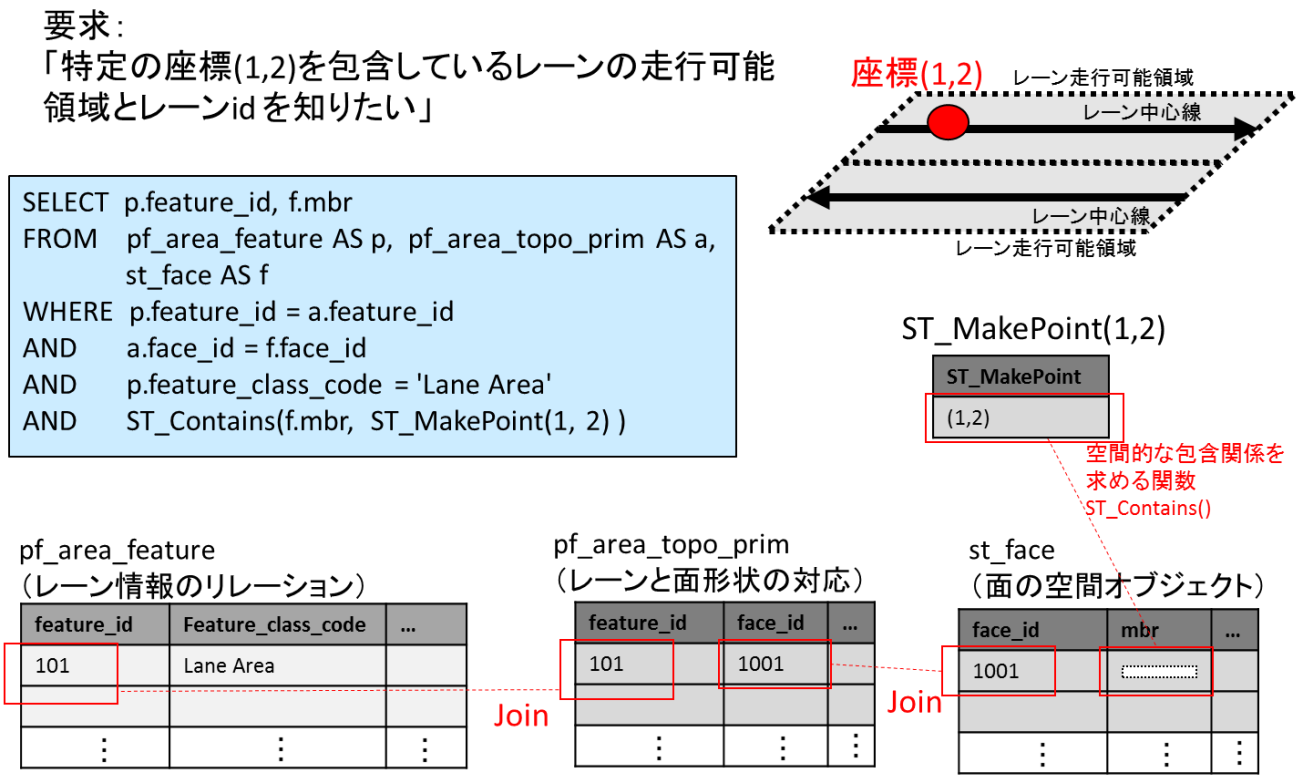


図 10　指定した点を包含するレーンの検索

5.3. 道路地図の検索(2)

図は「特定のレーン(例ではレーン ID=101)から、レーン間の経路接続関係を辿って到達できるレーンの ID を知りたい」という場合の one-shot query の記述例である。

FROM 節では 3 種類のテーブルが記述されている。relationship は名前の通り名古屋大学 COI 高精度地図フォーマットにおけるレーン間関係である Relationship を格納するテーブルである。relationship_feat は Feature(レーン)と Relationship の対応関係を保持するテーブルで、クエリでは接続元のレーンと接続先のレーンを探すために 2 回参照している。role_number と rel_type には本来数値のコードが入るが、文章の読みやすさのため文字で表記した。

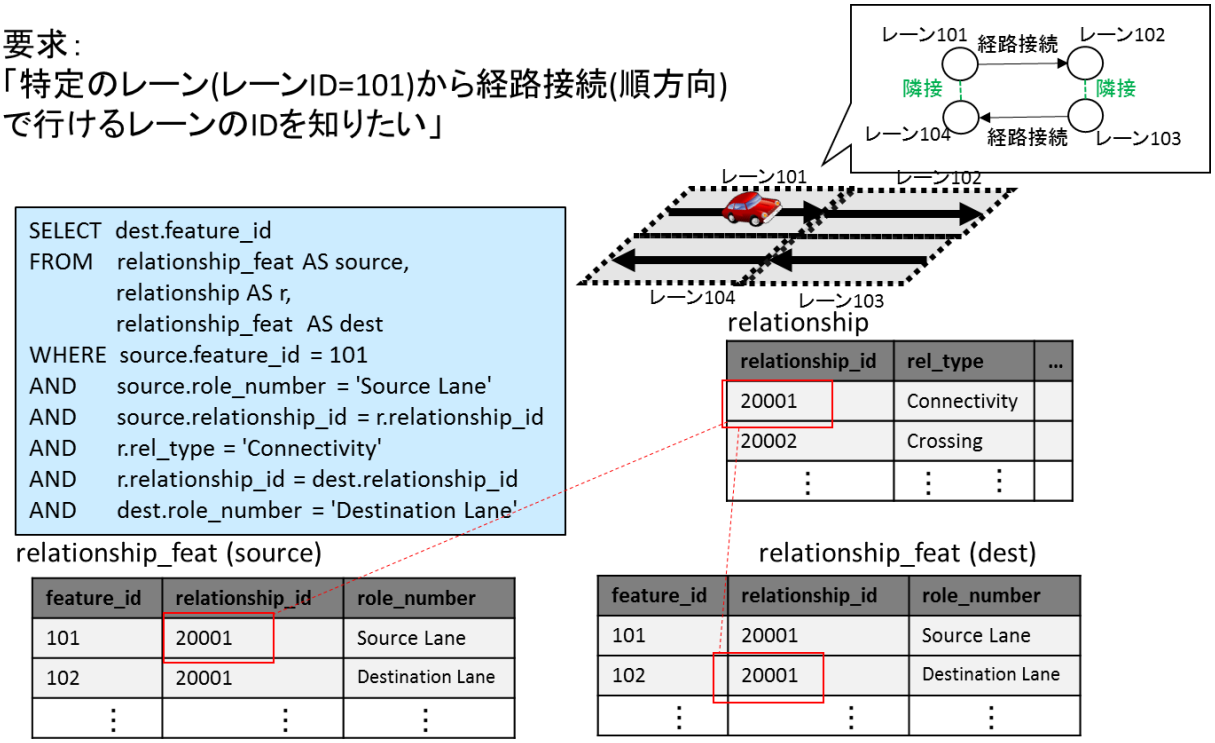


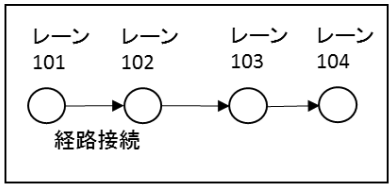
図 11　経路接続の関係を使った次レーンの検索

5.4. 道路地図の検索(3)

レーンネットワークに対する再帰検索の例を図に示す。図は「特定のレーン(例ではレーン ID=101)から経路接続(順方向)を 3 ホップで辿って到達できるレーンの ID を知りたい」という場合の one-shot query の記述例である。

WITH RECURSIVE が使われており、直後の括弧の中が繰返し実行される。UNION の前の部分と後の部分で分かれており、UNION の前の部分には再帰検索の初期集合（ここではレーン ID 101 だけの集合となる）を求めるサブクエリ、UNION の後の部分には初期集合の要素から経路接続関係を 1 ホップずつ辿って拡張していくためのサブクエリがかかれている。再帰検索の結果は multihop という名前のリレーションにためられていく。再帰の終了条件は WHERE 節の「multihop.hop < 3」であり、3 ホップ以上の探索はしないように設定されている。

要求:
「特定のレーン(レーンID=101)から経路接続(順方向)を3ホップ辿って到達できるレーンのIDを知りたい」




```
WITH RECURSIVE  multihop  AS
(
   SELECT  l.feature_id, 0  AS hop
   FROM    pf_comp_feature AS l
   WHERE  l.feature_id = 101
UNION
   SELECT  dest.feature_id, (multihop.hop + 1)
   FROM    multihop,
           relationship_feat  AS  source,
           relationship AS r,
           relationship_feat  AS  dest
   WHERE  multihop.feature_id = source.feature_id
   AND     multihop.hop < 3
   AND     source.role_number  = 'Source Lane'
   AND     source.relationship_id  = r.relationship_id
   AND     r.rel_type = 'Connectivity'
   AND     r.relationship_id = dest.relationship_id
   AND     dest.role_number  = 'Destination Lane'
)
SELECT feature_id
FROM  multihop
```

multihop (初期集合)

feature_id	hop
101	0

multihop (繰返し1回目)

feature_id	hop
101	0
102	1

multihop (繰返し2回目)

feature_id	hop
101	0
102	1
103	2

multihop (繰返し3回目)最終結果

feature_id	hop
101	0
102	1
103	2
104	3

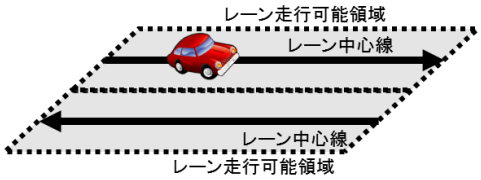
図 12　レーンに対する再帰検索

5.5. 動的情報と道路地図の統合

ここからは動的情報と地図情報を組み合わせた統合検索のためのクエリについて紹介する。図は「車両の動的情報をフィルタリングして車両 ID=0 の位置情報だけ取得し、その車両のいる座標を包含しているレーンの走行可能領域とレーン id を知りたい」という場合の continuous query の記述例である。

MASTER 節には車両の動的情報の vehicle_stream が指定されており、新規の車両の情報が到着する度にレーン検索も合わせて繰返し実行されることを意味している。

要求:
「自車両(id=0)のいる座標を包含しているレーンの走行可能領域とレーンidを知りたい」




```
MASTER  vehicle_stream
SELECT    v.id, v.position, p.feature_id, f.mbr
FROM      vehicle_stream AS v
          pf_area_feature AS p,
          pf_area_topo_prim AS a,  st_face AS f
WHERE   v.id = 0
AND       p.feature_id = a.feature_id
AND       a.face_id = f.face_id
AND       p.feature_class_code  = 'Lane Area'
AND       ST_Contains(f.mbr,  v.position )
```

vehicle_stream（車両の動的情報）

timeStamp	id	position	velocity	…
⋮	⋮	⋮	⋮	⋮
12:03:00	0	(1, 2)	56	…

空間的な包含関係を求める関数
ST_Contains()

pf_area_feature
（レーン情報のリレーション）

feature_id	Feature_class_code	…
101	Lane Area	
⋮	⋮	⋮

Join

pf_area_topo_prim
（レーンと面形状の対応）

feature_id	face_id	…
101	1001	
⋮	⋮	⋮

Join

st_face
（面の空間オブジェクト）

face_id	mbr	…
1001		
⋮	⋮	⋮

図 13　車両の位置座標を包含するレーン領域の検索

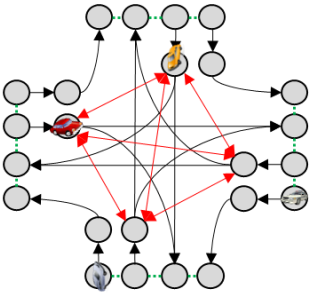
5.6. 道路地図を介した動的情報同士の統合

少し複雑なレーン関係の利用例として、「交差点で自車両と交差する可能性のある車両情報を提供してほしい」という場合を考える。目前に迫った緊急時の衝突回避は、センサを使ってお互いの相対位置を監視し、移動予測をしながら行われるものであるが、それよりも少し前の時点で対象となる候補をあらかじめ絞り込むという利用場面を想定している。図の例では「交差する可能性がある車両」をレーンの関係に置き換えて、自車両(ID=0)の位置情報からレーンを求め、そのレーンと「交差」関係にあるレーン上にいる車両を検索している。

ユーザ定義関数などの補助なしでは、のべ 11 個のテーブルの結合を指定しなければならないため、今後ユーザ定義関数などを用いて記述をさらに簡略化するように工夫が必要である。

```
MASTER  vehicle_stream
SELECT  v2.id, cross.feature_id
FROM    (
   SELECT v1.id, v1.position, p.feature_id
   FROM   vehicle_stream AS v1
             pf_area_feature AS p1,
             pf_area_topo_prim AS a1,  st_face AS f1
   WHERE   v1.id = 0
   AND       p1.feature_id = a1.feature_id
   AND       a1.face_id = f1.face_id
   AND       p1.feature_class_code = 'Lane Area'
   AND       ST_Contains(f1.mbr,  v1.position )
  ), (
   SELECT v2.id, v2.position, p.feature_id
   FROM  vehicle_stream AS v2,
             pf_area_feature AS p2,
             pf_area_topo_prim AS a2,  st_face AS f2
  WHERE   v2.id <> 0
   AND       p2.feature_id = a2.feature_id
   AND       a2.face_id = f2.face_id
   AND       p2.feature_class_code = 'Lane Area'
   AND       ST_Contains(f2.mbr,  v2.position )
   ),
   relationship_feat AS base,
   relationship AS  r,
   relationship_feat AS cross
WHERE   base.feature_id = p1.feature_id
AND       base.role_number = 'Base Lane'
AND       base.relationship_id = r.relationship_id
AND       r.rel_type = 'Crossing'
AND       r.relationship_id = cross.relationship_id
AND       cross.role_number = 'Crossing Lane'
AND       cross.feature_id = p2.feature_id
```

要求：
自車両(id=0)がいるレーンと交差するレーン上にいる他車両(id≠0)の情報を提供してほしい



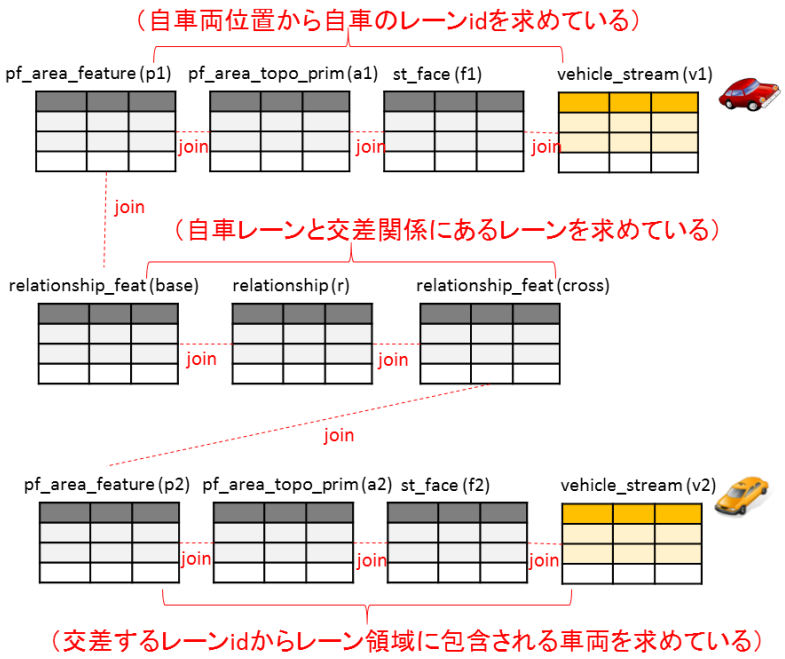


図 14　地図を介した動的情報同士の統合

終わりに

成果物を公開した目的は、ダイナミックマップのためのクエリ言語が備えるべき機能について、幅広い領域からフィードバックを得ることにある。本文書に関するご意見・質問は下記の窓口へご送信願いたい。

連絡先：
ダイナミックマップ 2.0 コンソーシアム事務局
〒464-8601 名古屋市千種区不老町 NIC 508
dm2-sec@nces.i.nagoya-u.ac.jp
https://www.nces.i.nagoya-u.ac.jp/dm2/index.html

謝辞

本書はダイナミックマップ 2.0 コンソーシアムの 2016 年度の成果物である。また、本研究の一部は、文部科学省「革新的イノベーション創出プログラム(COI STREAM)」の助成を受けている。