ORACLE

Oracle Spatialのパフォーマンスの最適化

ベスト・プラクティス、ヒントとコツ

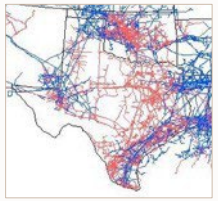
Daniel Geringer
空間ソリューション・スペシャリスト
2022年9月1日

Oracle Spatialの機能
すべてのOracle Databaseライセンスに組込み

デプロイ可能な
コンポーネント

マッピング | ジオコーディング | ルーティング | Webサービス(OGC) | Studio

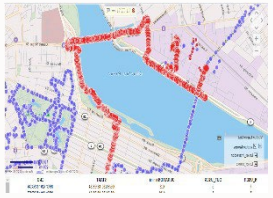

ポイント


行


ポリゴン


位置追跡
(ジオフェンシング)


ネットワーク

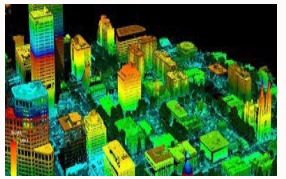

時空間
GPS追跡向け


住所ジオ
コーディング

線形参照


トポロジ


ラスター


3D / LiDAR

Oracle Spatial – 空間データおよびモデル

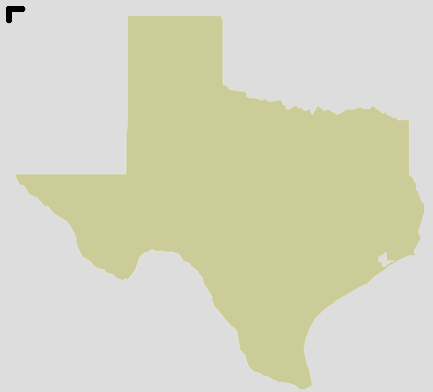
- 非空間データと同じ**セキュリティ、高可用性、管理性、データ整合性、およびスケーラビリティ**でデータベース表に保存されている空間データ

  - ベクター・データ – 点、線、ポリゴン
  - ラスター・データ – デジタル・イメージおよびグリッド・データ
  - GPS追跡データ – 同時追跡分析 / ジオフェンス分析向け
  - LIDARデータ – 点群 / LIDARデータ
  - ネットワーク・モデル – 運転時間 / 接続性分析

- 透過的データ暗号化、Data Redaction、Active Data Guard、レプリケーション、パラレル問合せなど

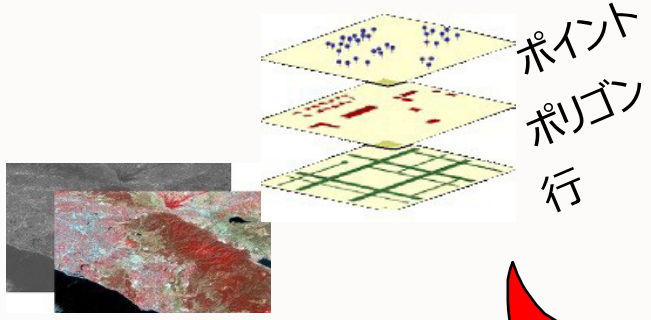
ベクター・データ

- 点、線、ポリゴン
- 通常のデータベース表に保存されるジオメトリ
- 通常のデータ・モデル概要
  - 正規化された表、1-1のリレーションシップ
  - 非正規化表は推奨されない
- ジオメトリの検証
- 空間の索引付け
- 空間問合せ – ほぼいつでも、空間条件はもっとも選択性が高い

STATE_NAME	CAPITAL	GEOMETRY
カリフォルニア	サクラメント	
テキサス	オースティン	

インデータベース空間機能

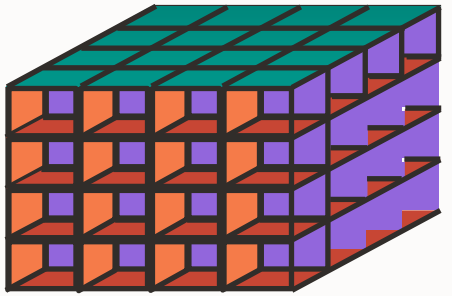
空間のデータタイプ



データベースに保存された
空間データ
（ベクター、ラスター、Lidar）

空間的に
有効化された
データベース

空間の索引付け



空間データへの
高速アクセス

SQLによる空間分析

```
SELECT a.customer_name, a.phone_number
FROM policy_holders a
WHERE sdo_within_distance ( a.geom, hurricane_path_geom,
            'distance = 10 unit = mile') = 'TRUE';
```

ベクター・データ – 表のパーティション化

- 通常は5,000万行超の表サイズの場合に推奨されるが、大幅に小さい表でも効果的な場合がある。
- **時間的なパーティション化**がごく一般的である。
  - 管理性向上の目的 – EXCHANGEパーティションおよびDROPパーティションにより、新しいデータの迅速な一括追加および古いデータのエージアウトが簡素化される。
  - パフォーマンス向上の目的 –
    - 指定した期間内で“のみ”検索を有効化する。
    - パーティション化されていない場合、最初に全期間にわたって空間の計算が適用され、次に時間条件が適用される。
- **機能タイプのパーティション化**も非常に効果的となる可能性がある。
  - たとえば、FEATURE_TYPE = transformer、substation、manhole、utility poleなどとする。
  - パーティション化がない場合、すべての機能に空間が適用されてから、feature_typeが適用される。これは最適ではない。
  - パーティション化により、関心のあるfeature_typeにのみ空間の検索が有効となる。

ベクター・データ – 表のパーティション化戦略

- レンジ、ハッシュ、リスト、時間隔、および参照パーティション化戦略
- コンポジット（パーティションごとにサブパーティションを生成）
  - レンジ-レンジ
  - レンジ-ハッシュ
  - レンジ-リスト
  - リスト-レンジ
  - リスト-ハッシュ
  - ハッシュ-ハッシュ
  - ハッシュ-リスト
  - ハッシュ-レンジ
- ローカル・パーティション空間索引は非常に効果的

ベクター・データ – ジオメトリの検証が重要

- Open Geospatial Consortium（OGC） – 標準ジオメトリ検証
- データセットに無効なジオメトリが含まれるのはきわめて一般的
- 一般的な問題 –
  - 線またはポリゴンで繰り返す連続した点
  - 自己交差するポリゴン
- 無効なジオメトリにより不正確な結果となる場合がある
- 組込み検証ルーチンを使用して無効なジオメトリを特定する（validate_geometry_with_context）
- 組込みルーチンを使用して無効なジオメトリを修正する（rectify_geometry）

ジオメトリを検証する最速の方法 – パラレル問合せを使用

- SDO_GEOM.VALIDATE_LAYER_WITH_CONTEXTに同様の出力
- 並列度を管理

```
CREATE TABLE validation_results PARALLEL 16 NOLOGGING AS
SELECT sdo_rowid, status
FROM (SELECT rowid sdo_rowid,
             sdo_geom.validate_geometry_with_context(geom, tolerance) status
      FROM roads)
WHERE status <> 'TRUE';
```

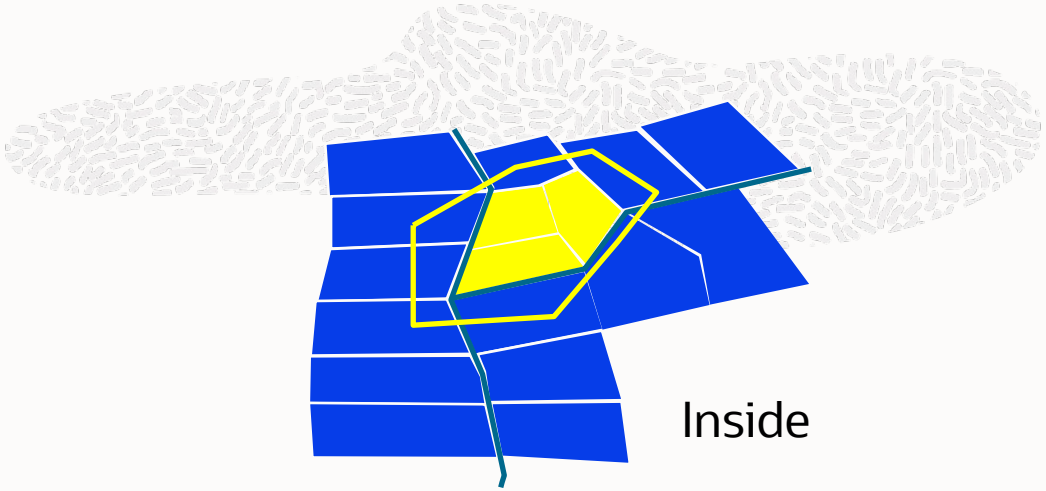
測地データの許容差 – 0.05未満もサポート

これが重要な理由

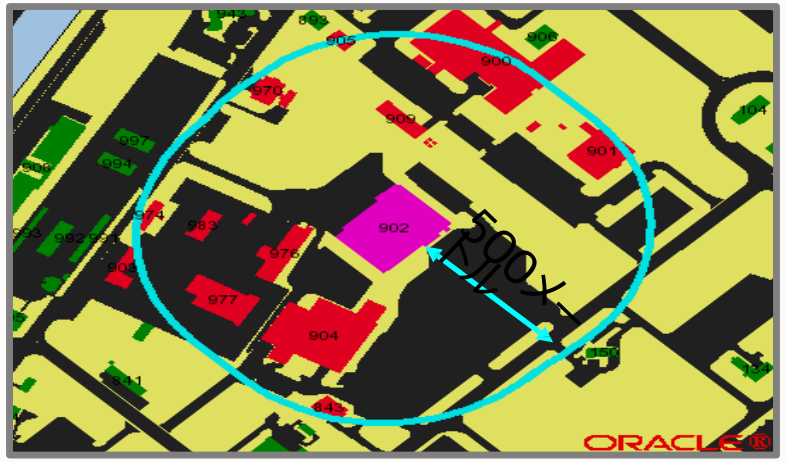
- ジオメトリは0.05の許容差で無効になる場合があり、その場合は修正が必要
- 修正前に、0.05（5センチメートル）よりも小さな許容差（例：0.005（5ミリメートル））を試行可能
- 許容差の変更のみによって、多くの13356（重複する頂点の繰返し）や13349（ポリゴンの自己交差）のエラーに対応可能
- 許容差は、比較を計画しているすべての空間レイヤー間で一貫していることが必要

ベクター・データ – 空間演算子

- 包括的な空間演算子
  - トポロジ演算子
    - Inside
    - Touch
    - Covers
    - Equal
    - Contains
    - Disjoint
    - Covered By
    - Overlaps
  - 距離演算子
    - Within Distance
    - Nearest Neighbor



Inside



Within Distance

空間ベクター・アクセラレーション

SPATIAL_VECTOR_ACCELERATION

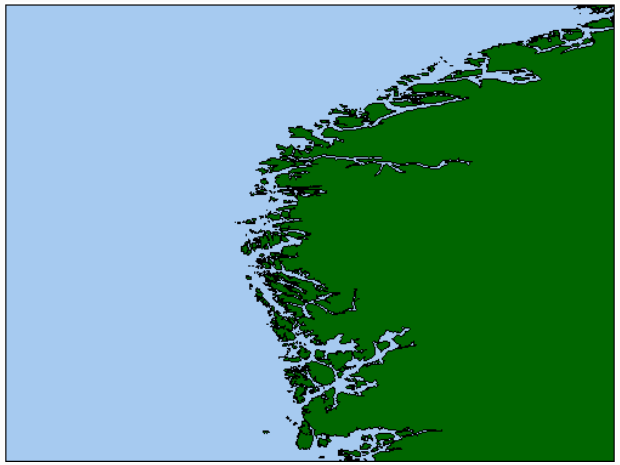
非常に重要な初期化パラメータ

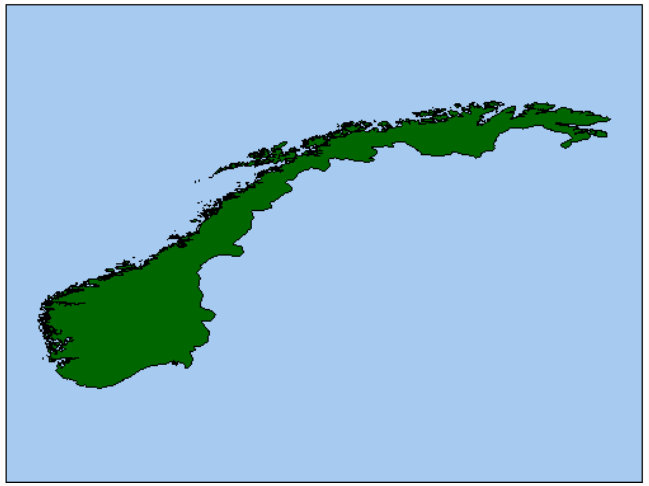
- 空間操作および空間機能のより**高速なアルゴリズム（数百倍高速）**
- ミッション・クリティカルな空間問合せパフォーマンス要件を含むすべてのアプリケーションに推奨
- Oracle初期化パラメータ – TRUEに設定されていることを確認
  - ALTER SYSTEM SET SPATIAL_VECTOR_ACCELERATION = TRUE
  - ALTER SESSION SET SPATIAL_VECTOR_ACCELERATION = TRUE
- あらゆるユーザーにメリット

SPATIAL_VECTOR_ACCELERATION

Oracle初期化パラメータ

- 空間演算子
  - **“high vertex count”**問合せウィンドウ（空間演算子の2番目の引数）のパフォーマンスの最適化
  - リレーション・マスクにより数百倍の高速化（すなわち、COVEREDBY、COVERS、TOUCHなど）
    - タイムゾーン・ポリゴンの例
    - きわめて詳細な海岸線
    - 343,395の頂点
    - 数百倍高速
    - このテストでは300倍高速





Oracle Support Note – Doc ID 2514624.1

12c以降のデータベース向けの最新のSpatialパッチ・バンドルは?

- ほとんどの場合に必要とされるSpatialパッチはDBRUに含まれる
- Oracle Supportが維持するライブ・ドキュメント
  - 適用を推奨されるDBRU特有のSpatialパッチ
  - 新しいDBRUのリリース時に更新
  - 適用するSpatialパッチが廃止された場合に更新
- 19.14 DBRU以降
  - ほとんどの場合に必要とされるSpatial更新はDBRUのみに含まれる
  - Doc ID 2514624.1 DBRU特有のSpatialパッチは次回のDBRUに含まれる予定であり、期限によっては、後続のDBRUになる可能性
- 19.13 DBRU以前では、Spatialパッチを適用するための鍵はDoc 2514624.1

ディスク上での空間データの体系化

パフォーマンスを最適化するための戦略

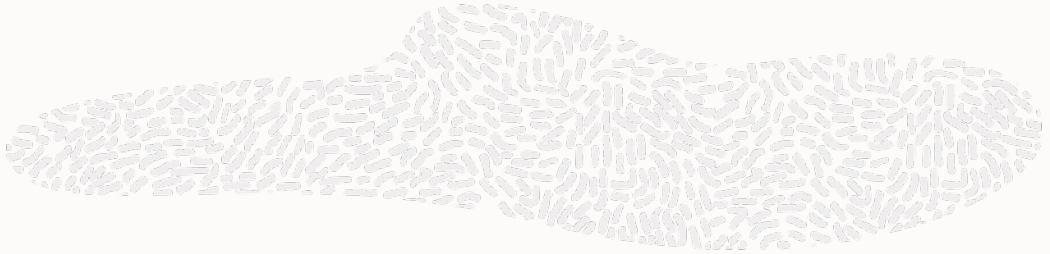
ディスク上での空間データの体系化

空間問合せパフォーマンスの最適化

- 空間索引が体系化されている（デフォルト）
  - 空間索引はセカンダリ表に保存され、Oracle（MDRT$表）によって管理される
  - 空間索引（ジオメトリMBRを格納）、および元表のジオメトリへの行IDポインタ
  - 空間索引によって同一のデータベース・ブロックで互いに近い位置にあるMBRがクラスタ化される
- 空間索引が体系化されていない（デフォルト）
  - ジオメトリMBRは同一のデータベース・ブロックでクラスタ化される一方、関連する元表のジオメトリは通常は分散される
  - 問合せ時に、分散したジオメトリによって多数のデータベース・ブロックが取得される可能性がある
  - 解決策は線形キーでの順序付けである

ディスク上での空間データの体系化

2つの戦略



- 点のみのデータの場合
  - Oracleの組込み機能を使用 – 属性クラスタリング
- 線およびポリゴンの場合
  - Oracle Spatialファンクション（sdo_util.linear_key）による線形キーでの順序付け
- 両方の戦略について、以降のスライドで説明します

空間データ – ディスク上での体系化

点のみのデータの場合 – 属性クラスタリングを使用

- インターリーブされた属性クラスタリング –
  - 空間に固有ではない
  - 点をSDO_GEOMETRYとしてではなく、2つのNUMBER列として格納する必要がある
  - ファンクション空間索引を作成できる
  - 時間、x、yもクラスタ化できる
- 点データが経度/緯度の場合
  - 以下の句をCREATE TABLE文に追加するのみ
  - CLUSTERING BY INTERLEAVED ORDER (longitude, latitude) YES ON LOAD;
  - 次のスライドで具体例を紹介します

点のみのデータの場合 – 属性クラスタリングを使用（空間特有ではない）

例

```
CREATE TABLE track_table (user_id          NUMBER,
                          capture_time     DATE,
                          longitude        NUMBER,
                          latitude         NUMBER,
                          date_as_number NUMBER) NOCOMPRESS NOLOGGING
                          CLUSTERING BY INTERLEAVED ORDER (capture_time, longitude,
latitude) YES ON LOAD;
```

```
-- ステージング表や外部表などからの直接パス挿入操作でのみ利用可能な属性クラスタリング
INSERT /*+ APPEND PARALLEL (8) */ INTO TRACK_TABLE
SELECT user_id, capture_time, longitude, latitude,
          capture_time – to_date('01-01-2019', 'MM-DD-YYYY')
FROM external_staging_table;
```

空間データ – ディスク上での体系化

線およびポリゴン・データの場合 – 空間クラスタリングを使用（sdo_util.linear_key）

- インターリーブされた属性クラスタリングは、線またはポリゴン、またはSDO_GEOMETRY列を含む表向けではない
- 線およびポリゴンの場合は、代わりにSpatialファンクションsdo_util.linear_keyを使用する
- sdo_util.linear_key –
  - 座標系のグリッディングに基づく
  - グリッドのすべてのセルに一意のキーが含まれる
  - 点を入力すると、ファンクションは点が入るセルに関連付けられた一意キーを返す
  - 線およびポリゴンの場合は、入力点を選択する（例：最初の点または中心点）
  - 挿入時に、線形キーによる順序付けによってディスク上の線およびポリゴンの空間データが最適な方法でクラスタ化される
- 次のスライドで具体例を紹介します

空間データ – 表領域での体系化

線およびポリゴン・データの場合 – 空間クラスタリングを使用（sdo_util.linear_key）

```
CREATE TABLE ship_tracks_ordered (col1 NUMBER, col2 NUMBER, geom SDO_GEOMETRY, id
        NUMBER);
```

```
INSERT /*+ APPEND PARALLEL (6) */ INTO ship_tracks_ordered NOLOGGING
WITH part1 AS (   select col1, col2,
                         geom,
                         sdo_geom.sdo_pointonsurface (geom,.005) first_point FROM
                  ship_tracks_not_ordered
SELECT col1, col2,
        geom,
        row_number() OVER (ORDER BY  sdo_util.linear_key       (p1.first_point.sdo_point.x,
                                                                p1.first_point.sdo_point.y,
                                                                -180,-90,180,90,22)) id
FROM part1 p1;
```

注 - xおよびyを含むsdo_util.linear_keyシグネチャは、Oracle 19.13以降で使用可能です。他のシグネチャは、19.13より前で使用可能です。

- あらゆるデータに対応するエクステントはすべて使用できます。経度/緯度を使用する場合（-180,-90,180,90）
- メルカトル図法を使用する場合（-21000000, -75000000, 21000000, 240000000）

ファンクション空間索引

SDO_GEOMETRY列を含まない表の場合

ファンクション空間索引

SDO_GEOMETRY列を含まない表の場合

- 均一なジオメトリの場合（すべての行に同数の頂点）
  - 点データ（x1, y1）
  - 2点による線（x1, y1, x2, y2）
  - ボックス・ポリゴン（min_x, max_x, min_y, max_y）
- 手順（前のスライドのtrack_tableについて以降のスライドで例を紹介）
  1. SDO_GEOMETRYを返すファンクションを作成する
  2. user_sdo_geom_metadataを移入する
  3. ファンクション空間索引を作成する
  4. 空間問合せを実行する

ファンクション空間索引

手順1 – SDO_GEOMETRYを返すファンクションを作成する

```
CREATE OR REPLACE FUNCTION get_geometry (lon NUMBER, lat NUMBER)
    RETURN sdo_geometry DETERMINISTIC PARALLEL_ENABLE AS
BEGIN
    IF lon IS NULL OR lat IS NULL
    THEN
        RETURN NULL;
    ELSE
        RETURN sdo_geometry(2001,4326, sdo_point_type(lon,lat,null),null,null);
    END IF;
END;
```

注　SDO_GEOMETRY（または任意のオブジェクト）を返すファンクションは、問合せパフォーマンスを最適化するためにDETERMINISTICが宣言される必要があります。

ファンクション空間索引

手順2 – user_sdo_geom_metadataを移入する

```
INSERT INTO user_sdo_geom_metadata VALUES (
        'TRACK_TABLE', 'SCOTT.GET_GEOMETRY(LONGITUDE,LATITUDE)',
        sdo_dim_array(sdo_dim_element('x',-180,180,.005),
                      sdo_dim_element('y',-90,90,.005)).
        4326);
```

- **注** user_sdo_geom_metadataの登録時は以下を実行する必要があります。
  - 列名ではなくファンクション名を指定する
  - OWNER.FUNCTION_NAMEを指定する
  - ファンクション・パラメータが表の列名に一致するようにする

ファンクション空間索引

手順3 – ファンクション空間索引を作成する

```
CREATE INDEX track_table_sidx ON track_table (get_geometry(longitude,latitude))
INDEXTYPE IS mdsys.spatial_index_V2 PARAMETERS('layer_gtype=point
        cbtree_index=true')
```

- **注**
  - RツリーおよびCBツリー空間索引がサポートされています。Rツリーの場合は、cbtree_index=trueを省略します。
  - CBツリー空間索引については、今後のスライドで詳しく説明します。
  - CBツリー空間索引では、パーティション化された表のローカル空間索引用にmdsys.spatial_index_V2が必要です。
  - 索引の作成時にlayer_gtype=pointを指定して、ポイントのみのレイヤーに対する問合せパフォーマンスを最適化します。

ファンクション空間索引

手順4 – 空間問合せを実行する

```
SELECT count(*)
FROM track_table
WHERE sdo_anyinteract (get_geometry(longitude,latitude),
                       sdo_geometry(2003,4326,null,sdo_elem_info_array(1,1003,3),
                                    sdo_ordinate_array(-75,35,-74,36)))='TRUE';
```

- **注**
  - 通常、ジオメトリ列が空間演算子の最初のパラメータとして指定される
  - 代わりに、ファンクション空間索引の作成に使用されるファンクションを指定する

CBツリー – ポイントのみの空間索引

ポイント・データのストリーミングのために最適化

CBツリー – ポイントのみの空間索引

- 空間索引を有効にしてストリーミングされたポイント・データを取り込むために最適化
- CBツリー空間索引
  - 複数のセッションから同時DMLに対応できるように設計（すなわち、接続プール）
  - ごく短時間で空間索引を作成
- 空間機能の侵害なし
- cbtree_index=trueを指定

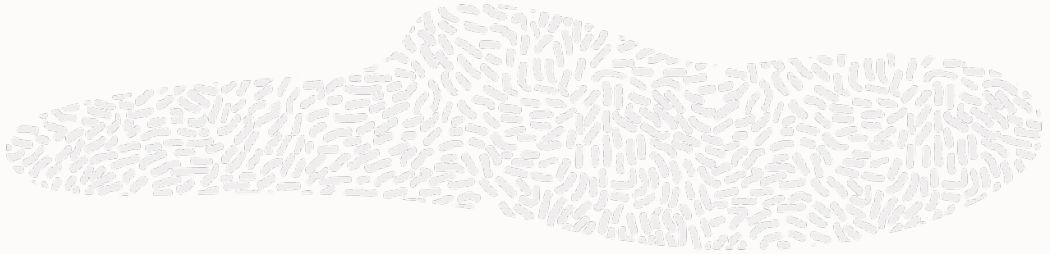
```
CREATE INDEX point_sidx ON cities (geometry)
INDEXTYPE IS mdsys.spatial_index_v2
PARAMETERS('layer_gtype=point cbtree_index=true');
```

パラレル問合せと空間

US Railアプリケーション

パラレル問合せと空間演算子

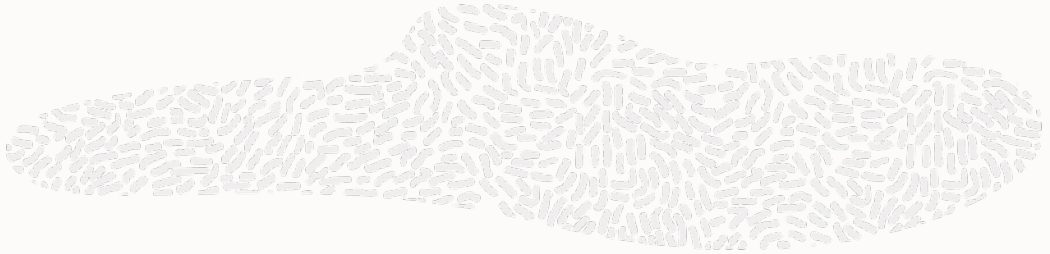
US Railアプリケーション



- 要件
  - 1日を通じて各列車のGPSの位置が収集されること
  - 位置ごとに他の属性が含まれること（時間、速度など）
  - GPSの位置にはある程度のエラーが含まれるため、常に線路上に位置するとは限らない
  - 最近傍問合せを一括してもっとも近い線路を見つけ、レポートされた列車の位置を線路上に投影
- この情報の使用目的
  - 列車の追跡
  - メンテナンスの分析、エンジニアがパラメータ内にいることの確認など

パラレル問合せと空間演算子

テスト内容



- 45,158,800のGPS列車位置
- 列車の位置ごとに以下を実行
  - 列車にもっとも近い線路を検索（SDO_NNを使用）
  - 続いて、列車にもっとも近い線路上の位置を計算

パラレル問合せと空間演算子

US Railアプリケーション

```
CREATE TABLE results PARALLEL 72 NOLOGGING AS
SELECT a.locomotive_id, sdo_lrs.find_measure (b.track_geom, a.locomotive_pos) FROM
locomotives a, tracks b
WHERE sdo_nn (b.track_geom, a.locomotive_pos, 'sdo_num_res=1') = 'TRUE';
```

パラレル問合せと空間演算子

Exadataの結果

- Exadata Half RAC：
  - 34.75時間の順次実行と44.1分のパラレル実行
  - 線形スケーラビリティ - 48のデータベース・コア - 47倍高速
- 新世代のチップを搭載したX9-2でさらに高速 – 容易に100倍以上高速

空間クラスタリング

傾向分析のために

サーバー側のパラレル対応クラスタリング

- **傾向分析** - テレマティクス・クラスタリング（実に強力）
  - 数十億単位で収集されるGPSポイント
  - ポイントをクラスタ化して分析用にはるかに管理しやすいデータセットを生成
  - クラスタ化されたデータに関連付けられたパターンや傾向を特定
  - 1日の特定の時間帯のクラスタは特定の種類の店やレストランの近くに存在する傾向

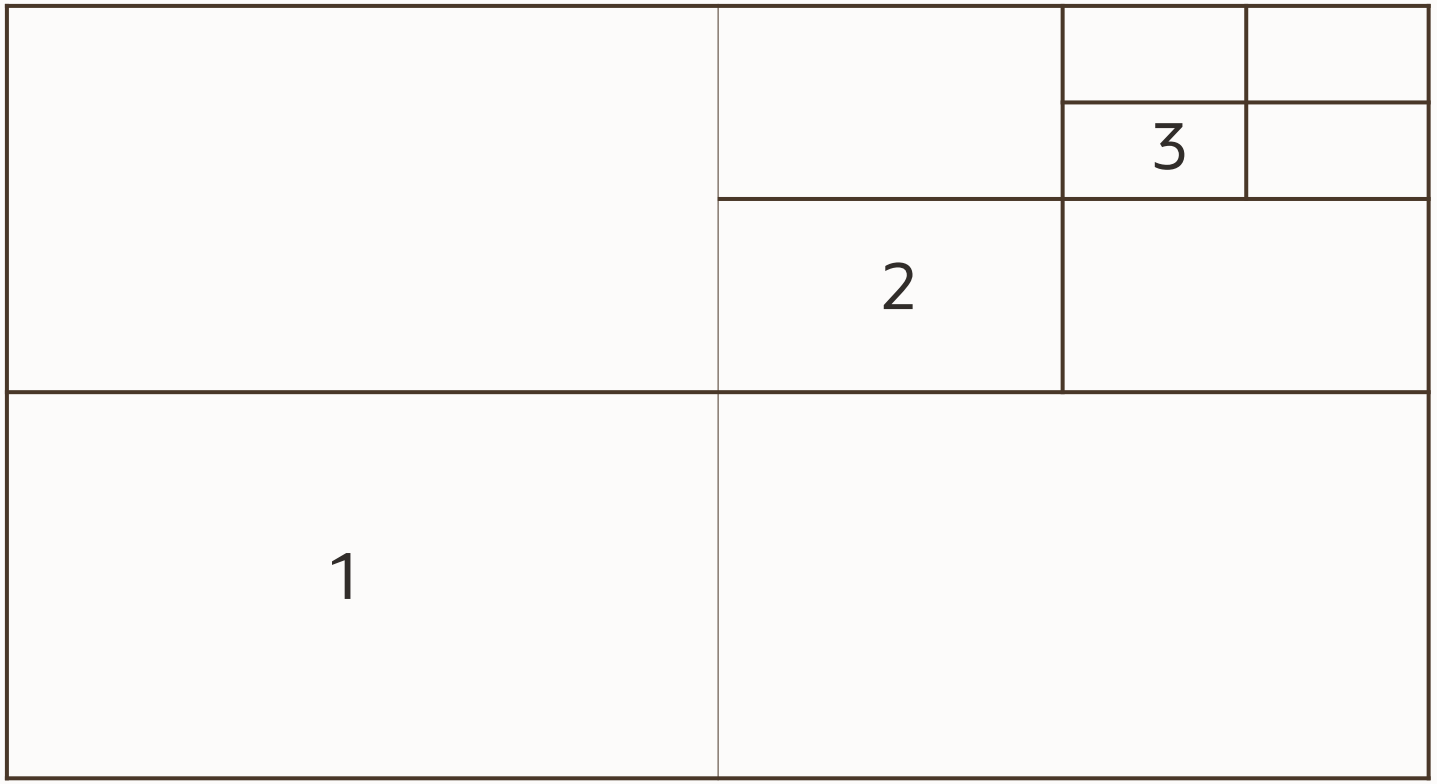
サーバー側のパラレル対応クラスタリング

- 数百万の行を数秒でクラスタ化（サーバー側）
  - 100万のポイントを0.86秒で62708のクラスタへ（パラレル16）（1秒未満のパフォーマンス）
  - 10億を超えるポイント（1,024,000,000）を7分で62708のクラスタへ（パラレル16）
- クラスタの中心と数を返す
- マッピング・アプリケーションの自動ズーム・イン/アウト・クラスタリングに効果的
- 特にクライアント側でクラスタ化する行が多すぎる場合
- 特に数百万や数十億のレコードをクラスタ化する場合、クラスタ化の結果は持続可能（事前計算済み）
- クラスタ化は即座に実行でき、パラレルにも対応

空間クラスタリング – sdo_util.linear_keyも使用

セル・サイズを選択 – クアッド・タイル・ベース

- レベルはsdo_util.linear_keyのパラメータ
- クラスタリング用のタイル・サイズを定義
- レベル1 – 1/4座標系
- レベル2 – 1/16座標系
- レベル3 – 1/64座標系
- など

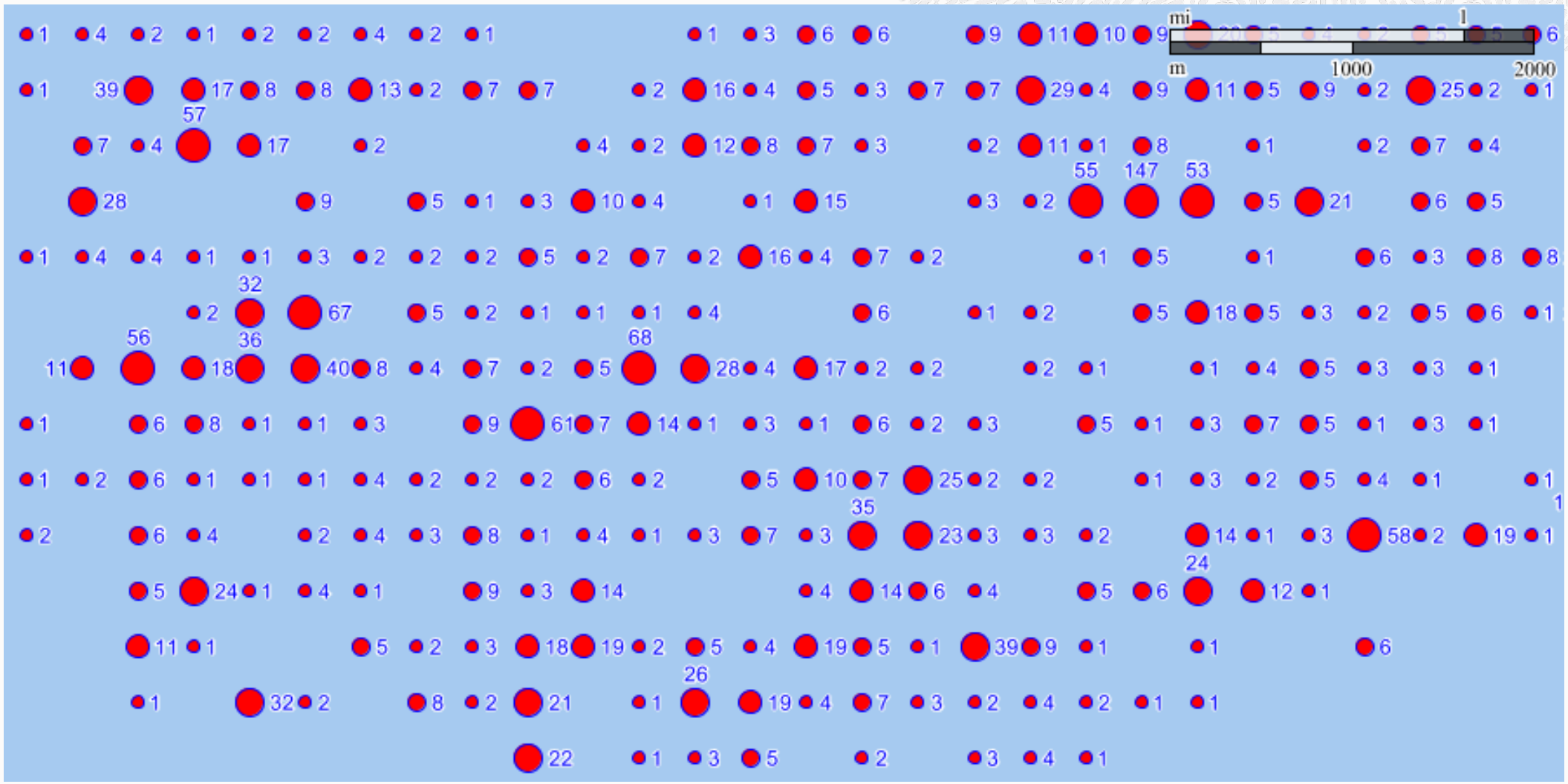


空間クラスタリング - 例

```
ALTER SESSION ENABLE PARALLEL DML;
CREATE TABLE results (cnt NUMBER, center SDO_GEOMETRY);
INSERT /*+ append parallel(16) */ INTO results NOLOGGING
SELECT count(*),
       sdo_util.linear_key_center (cell_id, -180, -90, 180, 90)
FROM ( SELECT sdo_util.linear_key (longitude, latitude, -180, -90, 180, 90, 15) as cell_id
       FROM one_billion_row_table a)
GROUP BY cell_id;
```

注　セルのジオメトリを表示させるにはsdo_util.linear_key_boundaryを使用します。シグネチャはsdo_util.linear_key_centerと同様です。

サーバー側の空間クラスタリング – 道路網 - 結果



空間クラスタリング – GPSデータの例

- 多数のユーザーのGPS位置をクラスタ化する場合、1人のユーザーによってクラスタ内の多数の場所がレポートされる場合がある
- 下記の例により、クラスタ数に“個別”ユーザーの数が確実に反映される

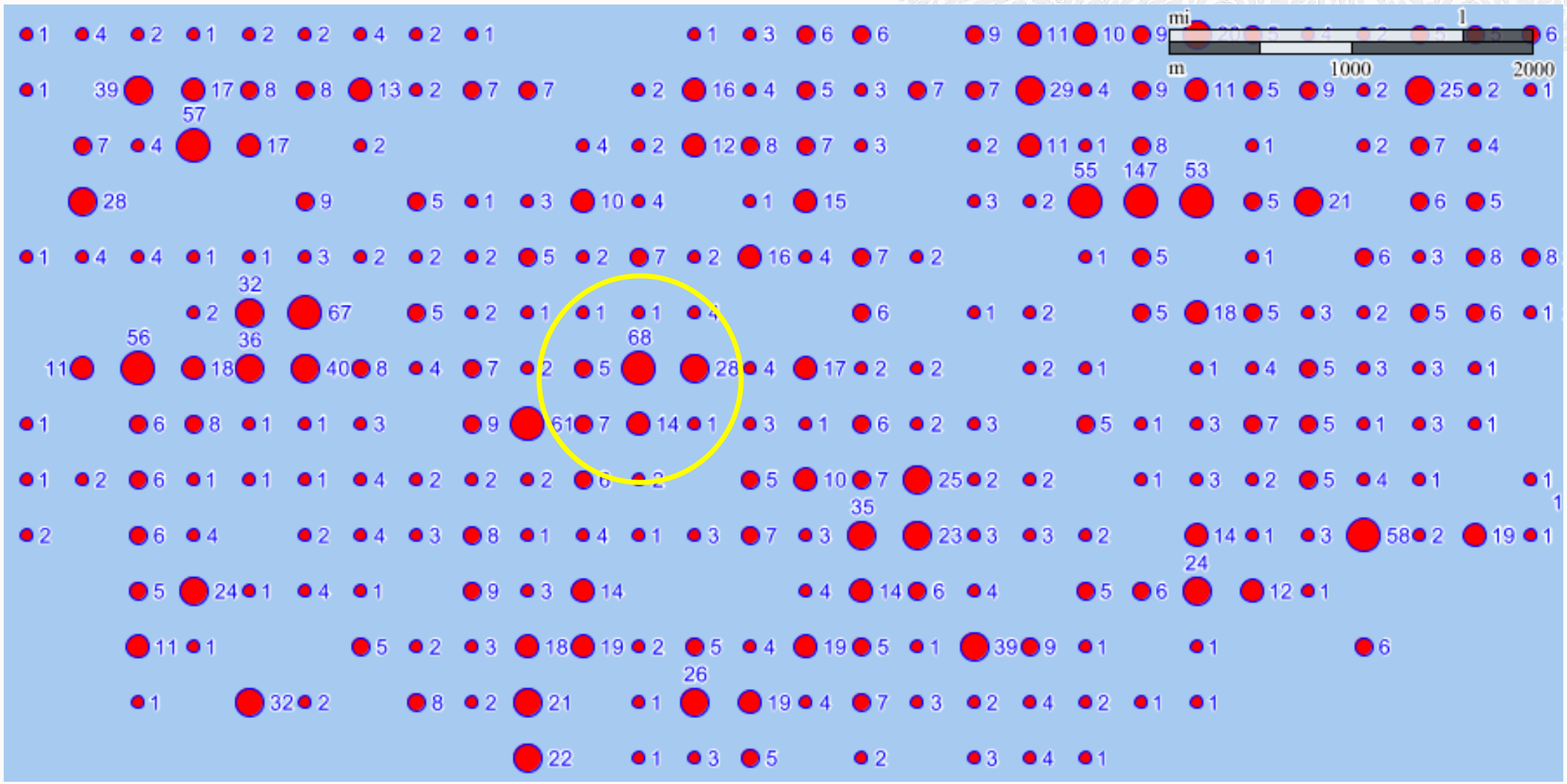
```
ALTER SESSION ENABLE PARALLEL DML;
CREATE TABLE results (cnt NUMBER, center SDO_GEOMETRY);

INSERT /*+ append parallel(16) */INTO results NOLOGGING
SELECT count(*), sdo_util.linear_key_center (cell_id, -180, -180, 180, 180)
FROM (SELECT cell_id, user_id, count(*)
      FROM (SELECT sdo_util.linear_key (longitude, latitude,-180,-90,180,90, 15) as
                   cell_id, user_id
            FROM one_billion_row_table a)
      GROUP BY cell_id, user_id )
GROUP BY cell_id;
```

時空間クラスタリング – GPSデータの例

- 多数のユーザーのGPS位置をクラスタ化する場合、1人のユーザーによってクラスタ内の多数の場所がレポートされる場合がある
- 下記の例により、クラスタ数に“個別”ユーザーの数が確実に反映される

```
ALTER SESSION ENABLE PARALLEL DML
CREATE TABLE results (day varchar2(100), hour_range_id NUMBER, cnt NUMBER, center SDO_GEOMETRY);
INSERT /*+ append parallel(16) */ INTO results NOLOGGING
SELECT day, hour_range_id, count(*) cnt,
          sdo_util.linear_key_center(cell_id, -180, -90, 180, 90, 15) center_geom
FROM (SELECT cell_id, user_id, day, hour_range_id, count(*)
        FROM (SELECT cell_id,
                     user_id,
                     day,
                     CASE WHEN hour_of_day >= 0 AND hour_of_day < 6 THEN 1
                          WHEN hour_of_day >= 6 AND hour_of_day < 10 THEN 2
                          WHEN hour_of_day >= 10 AND hour_of_day < 16 THEN 3
                          WHEN hour_of_day >= 16 AND hour_of_day < 20 THEN 4
                          WHEN hour_of_day >= 20 AND hour_of_day < 24 THEN 5
                     END hour_range_id
              From   (SELECT sdo_util.linear_key (lon, lat, -180,-90,180,90, 15) as cell_id, user_id,
                             to_char(reported_time, 'MONDDYYYY') day,
                             to_number(to_char(reported_time, 'HH24')) hour_of_day
                      FROM one_billion_row_table a))
       GROUP BY cell_id, user_id, day, hour_range_id)
GROUP BY cell_id, day, hour_range_id;
```

数が多い午前8時のクラスタの中心から2 km以内のすべてのビジネスを検索 – 傾向分析



Oracle Spatial – 空間データおよびモデル（サマリー）

- 非空間データと同じ**セキュリティ、高可用性、管理性、データ整合性、およびスケーラビリティ**で
  データベース表に保存されている空間データ

  - ベクター・データ – 点、線、ポリゴン
  - ラスター・データ – デジタル・イメージおよびグリッド・データ
  - GPS追跡データ – 同時追跡分析 / ジオフェンス分析向け
  - LIDARデータ – 点群 / LIDARデータ
  - ネットワーク・モデル – 運転時間 / 接続性分析

- 透過的データ暗号化、Data Redaction、Active Data Guard、レプリケーション、パラレル問合せなど

Oracle Spatialテクノロジーの参考資料

- Oracle Spatialテクノロジー：https://www.oracle.com/database/spatial/
- Oracle LiveLabs：https://bit.ly/golivelabs-spatial
- ブログ：https://blogs.oracle.com/oraclespatial/、
  https://blogs.oracle.com/database/category/db- spatial
- Slack（#spatial channelにご参加ください）：https://bit.ly/Join-ANDOUC-Slack
- YouTube：https://bit.ly/Spatial-Graph-YouTube
- AskTOMビデオ・シリーズ：https://bit.ly/AskTOMSpatial
- LinkedIn：https://bit.ly/Spatial-Graph-LinkedIn
- Twitter：@SpatialHannes、@Jeanlhm

Q&A

Zoom Q&Aボックスに質問を入力してください


ORACLE