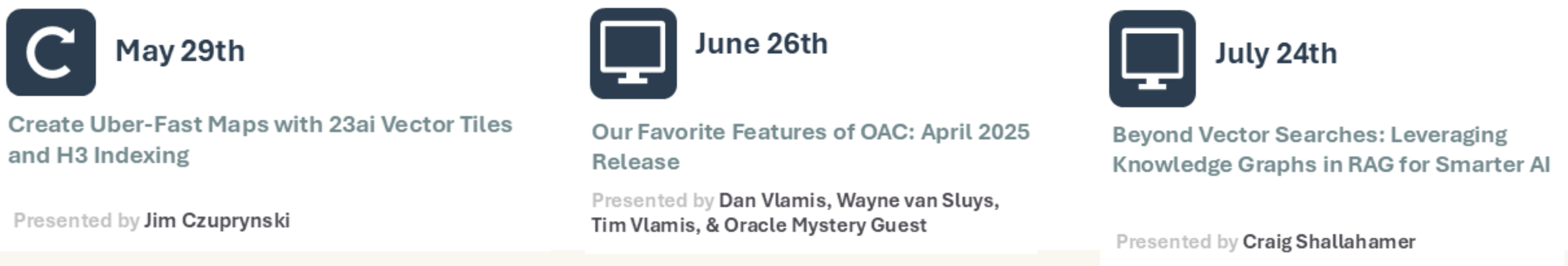
Create Uber-Fast Maps with 23ai Vector Tiles and H3 Indexing

Jim Czuprynski
Chief StoryTeller
Zero Defect Computing, Inc.

Future & Past TechCasts:



May 29th

Create Uber-Fast Maps with 23ai Vector Tiles and H3 Indexing

Presented by **Jim Czuprynski**

June 26th

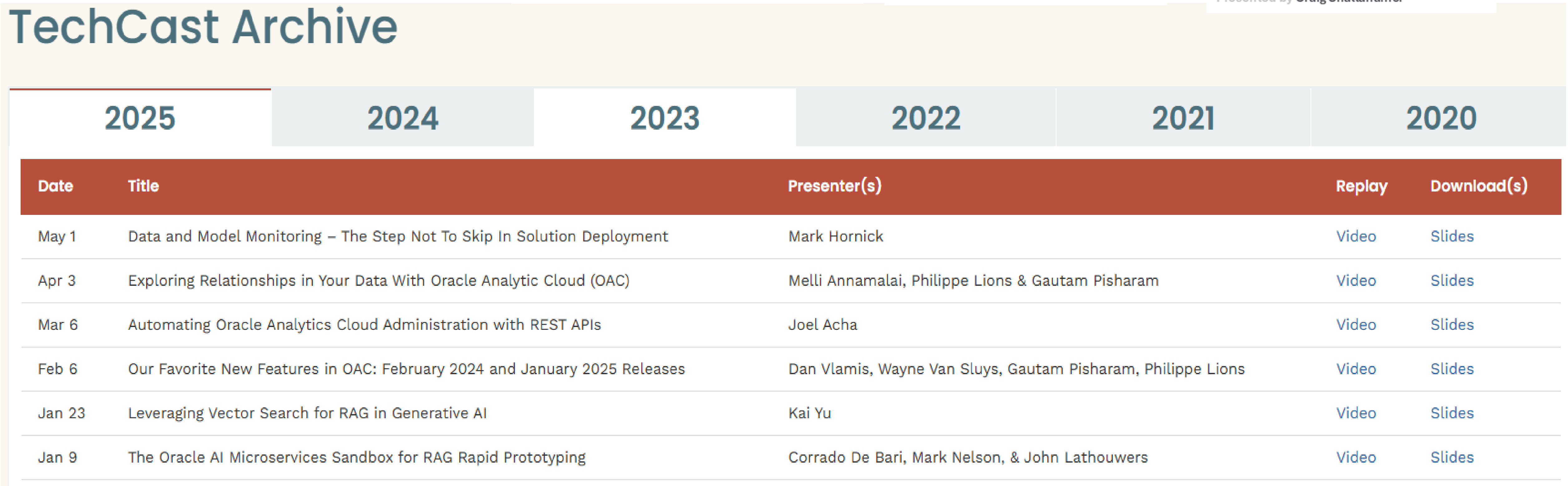
Our Favorite Features of OAC: April 2025 Release

Presented by **Dan Vlamis, Wayne van Sluys, Tim Vlamis, & Oracle Mystery Guest**

July 24th

Beyond Vector Searches: Leveraging Knowledge Graphs in RAG for Smarter AI

Presented by **Craig Shallahamer**



TechCast Archive

Date	Title	Presenter(s)	Replay	Download(s)
May 1	Data and Model Monitoring – The Step Not To Skip In Solution Deployment	Mark Hornick	Video	Slides
Apr 3	Exploring Relationships in Your Data With Oracle Analytic Cloud (OAC)	Melli Annamalai, Philippe Lions & Gautam Pisharam	Video	Slides
Mar 6	Automating Oracle Analytics Cloud Administration with REST APIs	Joel Acha	Video	Slides
Feb 6	Our Favorite New Features in OAC: February 2024 and January 2025 Releases	Dan Vlamis, Wayne Van Sluys, Gautam Pisharam, Philippe Lions	Video	Slides
Jan 23	Leveraging Vector Search for RAG in Generative AI	Kai Yu	Video	Slides
Jan 9	The Oracle AI Microservices Sandbox for RAG Rapid Prototyping	Corrado De Bari, Mark Nelson, & John Lathouwers	Video	Slides

Submit a topic to share at **https://andouc.org/techcasts/**

Analytics and Data
ORACLE USER COMMUNITY

Let’s Connect

Analytics and Data
ORACLE USER COMMUNITY
TechCast Days
WINTER SESSION
February 16–18, 2021

Website
http://andouc.org/

Chat with the Experts
https://bit.ly/Join-ANDOUC-Slack

Watch Previous TechCasts
https://bit.ly/3qmGgHN

https://www.linkedin.com/company/analytics-and-data-oracle-user-community

Spatial + Graph SIG
bit.ly/Spatial-Graph-LinkedIn

https://www.facebook.com/AnDOracleUserCommunity

@analyticsndataouc.bsky.social

@AnalyticAndData

ANALYTICS AND DATA
TechCasts

Who Am I, and What Am I Doing Here?

Traveler & public speaker

Winters: Illinois

Summers: Wisconsin

Avid amateur bird watcher

ORACLE ACE Director

ORACLE Certified Professional

Oracle Database Upgrade, Migration & Transformation Tips & Techniques
Oracle Database Problem Solving and Troubleshooting Handbook
Oracle Database 12c Release 2 Testing Tools and Techniques for Performance Stability
PDB Me to Oracle Cloud Pocket Solutions Guide

Oldest dude in martial arts class

Cyclist

XC skier

- ➢E-mail me at jim@jimthewhyguy.com
- ➢Follow me on BlueSky **(@ jimthewhyguy.bsky.social)**
- ➢Connect with me on LinkedIn (**Jim Czuprynski**)

Oracle ACE

The Oracle ACE Program

400+ technical experts helping peers globally

- The Oracle ACE Program recognizes and rewards community members for their technical and community contributions to the Oracle community
- 3 membership levels: Director, Pro, and Associate
- Nominate yourself or a colleague at ace.oracle.com/nominate
- Learn more at ace.oracle.com

✉ aceprogram_ww@oracle.com Facebook.com/OracleACEs @oracleace Oracle ACE Program Group

“There Is No Planet B”: Shifting Priorities …

Wind and Solar Overtake Coal Power for First Time in U.S.

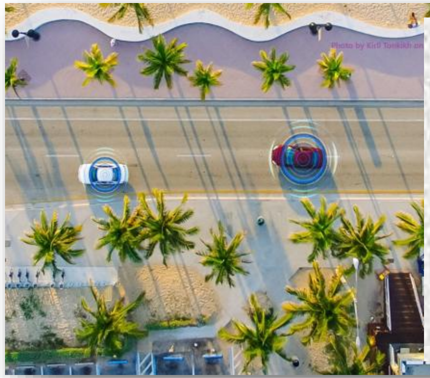
- *Wall Street Journal*, 13 March 2025

Overall, the U.S. installed 50 gigawatts of new solar capacity in 2024. … Wind and solar ***have overtaken coal in 24 states***, according to Ember, with Illinois the latest to join the ranks in 2024, following Arizona, Colorado, Florida and Maryland in 2023.

California now has 48% more EV chargers than gasoline nozzles in the state

- *State of California*, 20 March 2025

In a significant milestone for the state, California now has ***48% more public and shared private EV chargers*** than the number of gasoline nozzles. The California Energy Commission (CEC) estimates there are about 120,000 gas nozzles in the state, compared to 178,000 public and shared private chargers.



Study: EV charging stations boost spending at nearby businesses

- *MIT News*, 4 September 2024

“The positive impacts of EV charging stations on businesses ***are not constrained solely to some high-income neighborhoods***,” Wang says. “It highlights the importance for policymakers to develop EV charging stations in marginalized areas, because they not only foster a cleaner environment, but also serve as a catalyst for enhancing economic vitality.”

Business Case: Where Do We Locate All These Resources?

Solar, wind, and **nuclear** power generation strategies offer **wildly divergent** capabilities

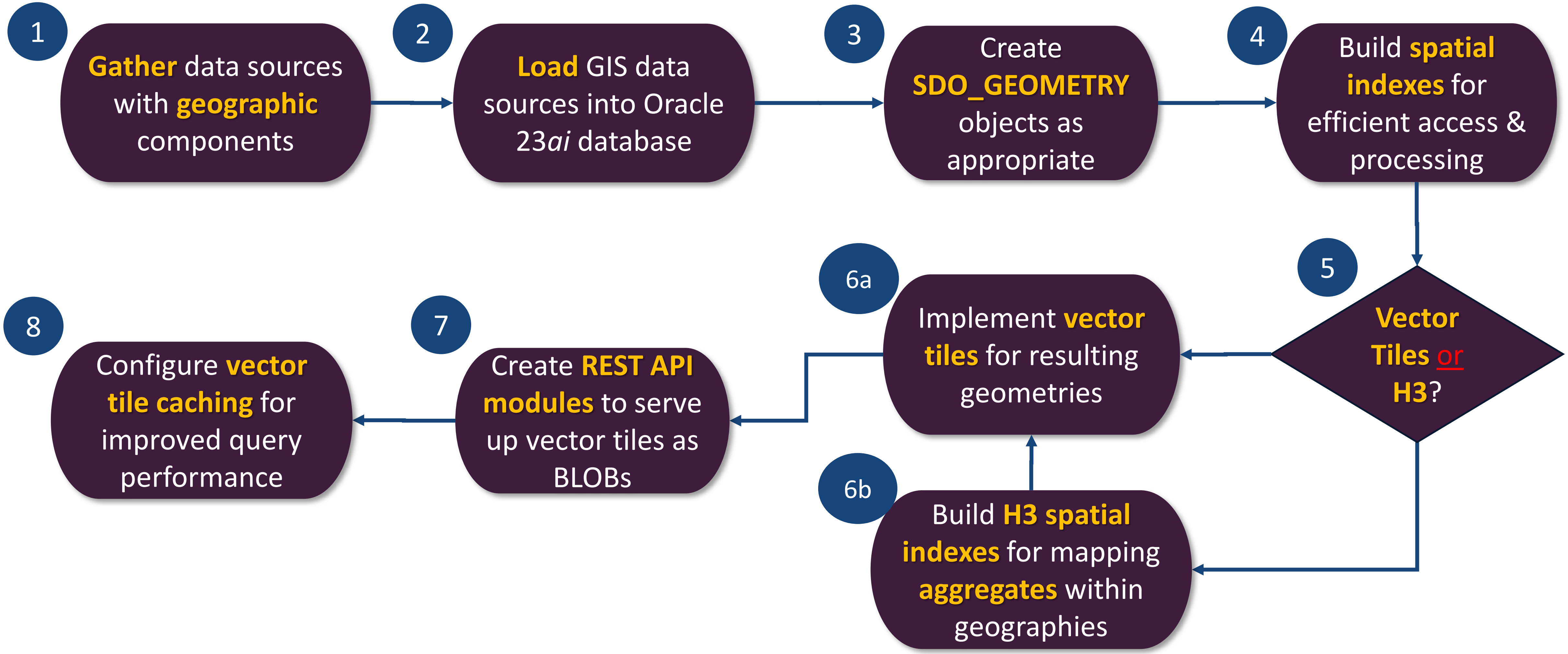
Current and future alternative energy production sites are **unevenly distributed** across the USA

Solar and **wind** are **weather-dependent** … so reliable **battery storage systems** to capture excess power should be centrally-located

Storage systems should be built near **existing and future electrical grid access points** to minimize power loss over long distances

Photo by Roger Starnes Sr on Unsplash

Steps for Implementing Vector Tiles and H3 Indexing



1 **Gather** data sources with **geographic** components

2 **Load** GIS data sources into Oracle 23*ai* database

3 Create **SDO_GEOMETRY** objects as appropriate

4 Build **spatial indexes** for efficient access & processing

5 **Vector Tiles** or **H3**?

6a Implement **vector tiles** for resulting geometries

6b Build **H3 spatial indexes** for mapping **aggregates** within geographies

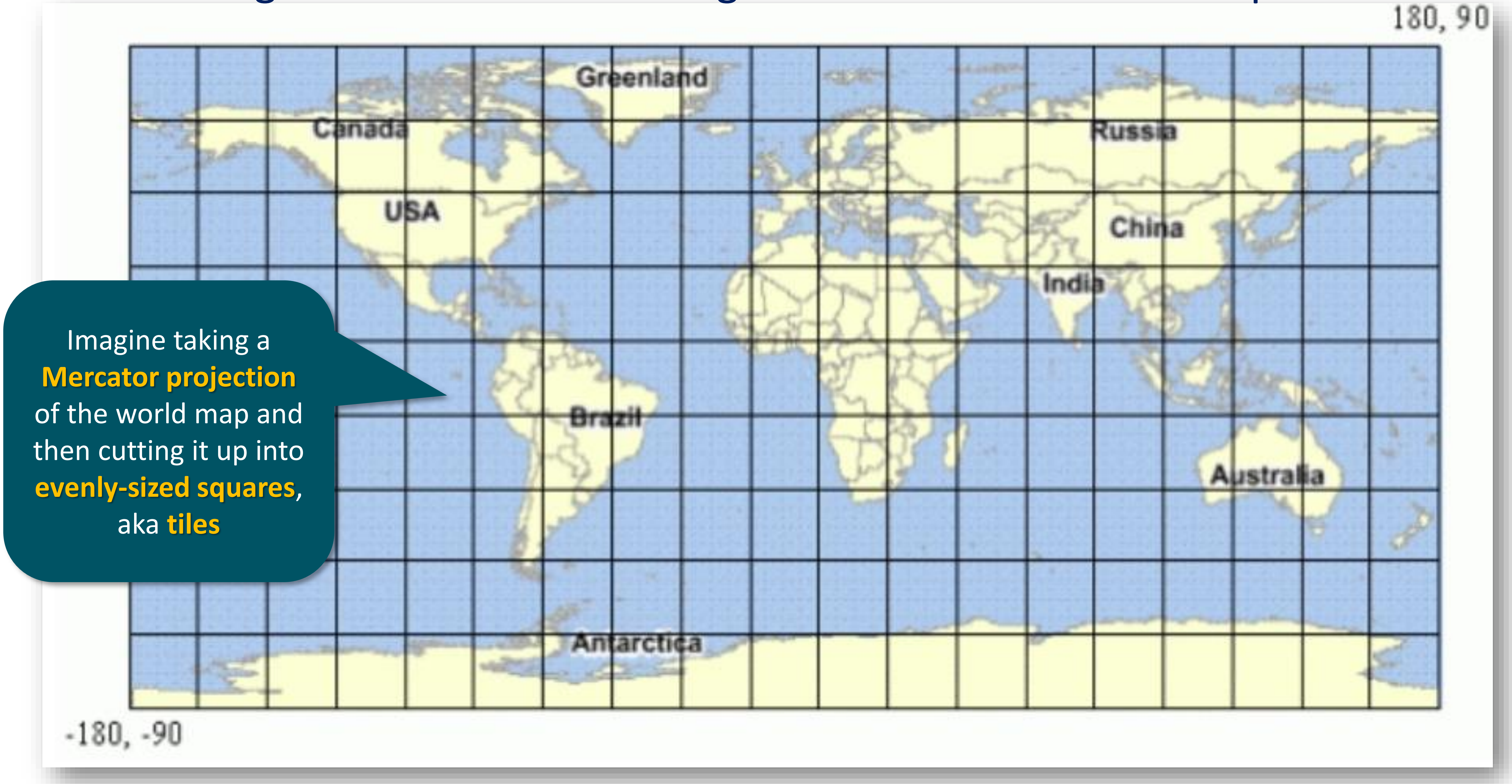
7 Create **REST API modules** to serve up vector tiles as BLOBs

8 Configure **vector tile caching** for improved query performance

Vector Tiles

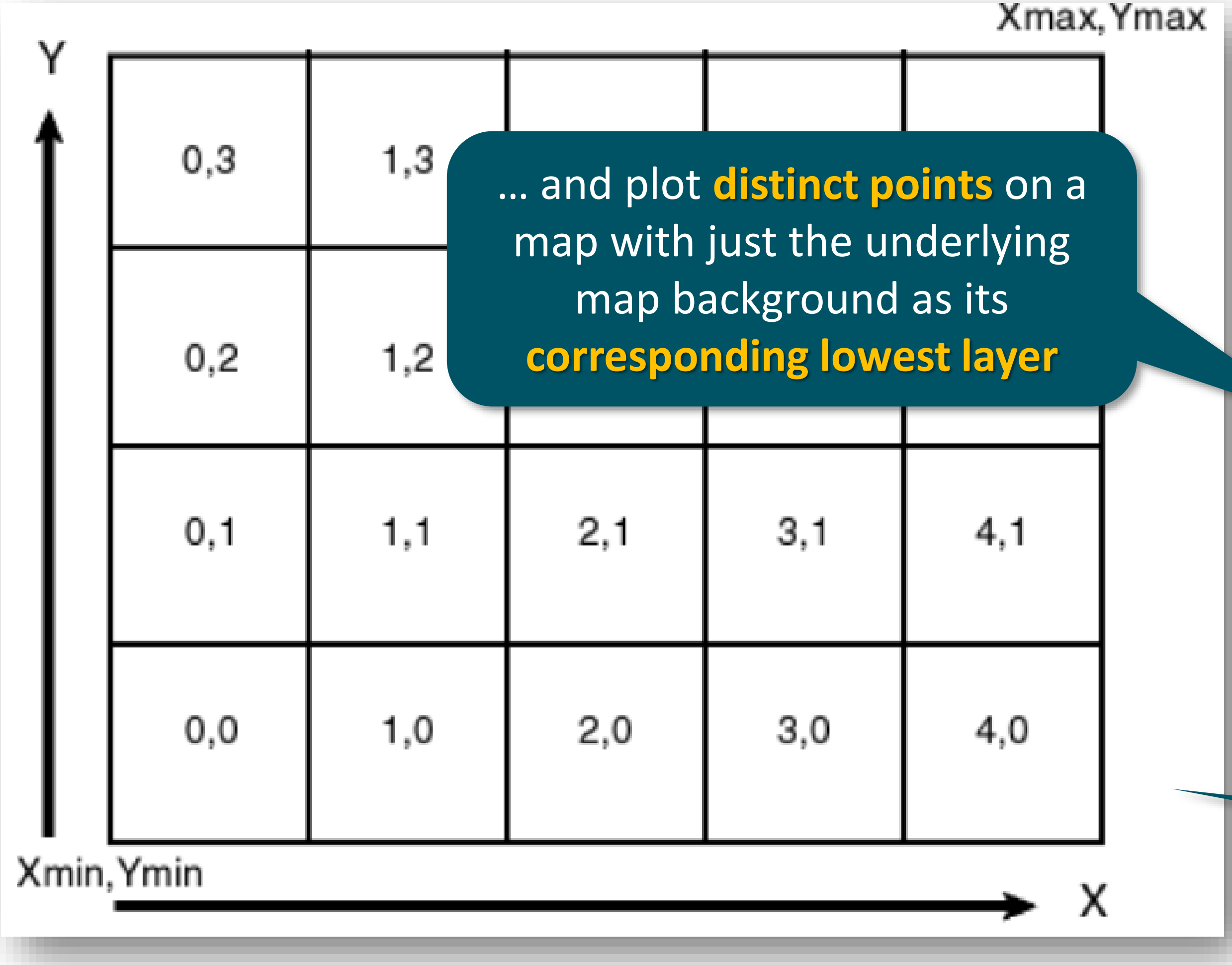
Photo by Brandon Griggs

Understanding Vector Tiles: Visualizing a Series of Ever-Smaller Squares

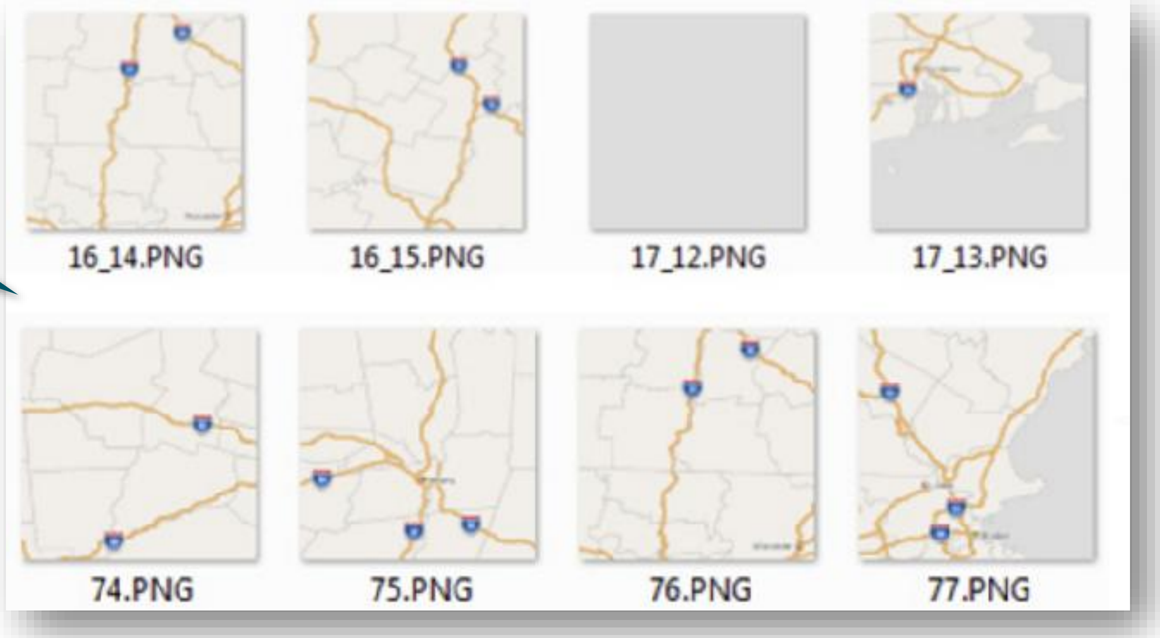


Imagine taking a **Mercator projection** of the world map and then cutting it up into **evenly-sized squares**, aka **tiles**

Understanding Vector Tiles: Visualizing a Series of Ever-Smaller Squares

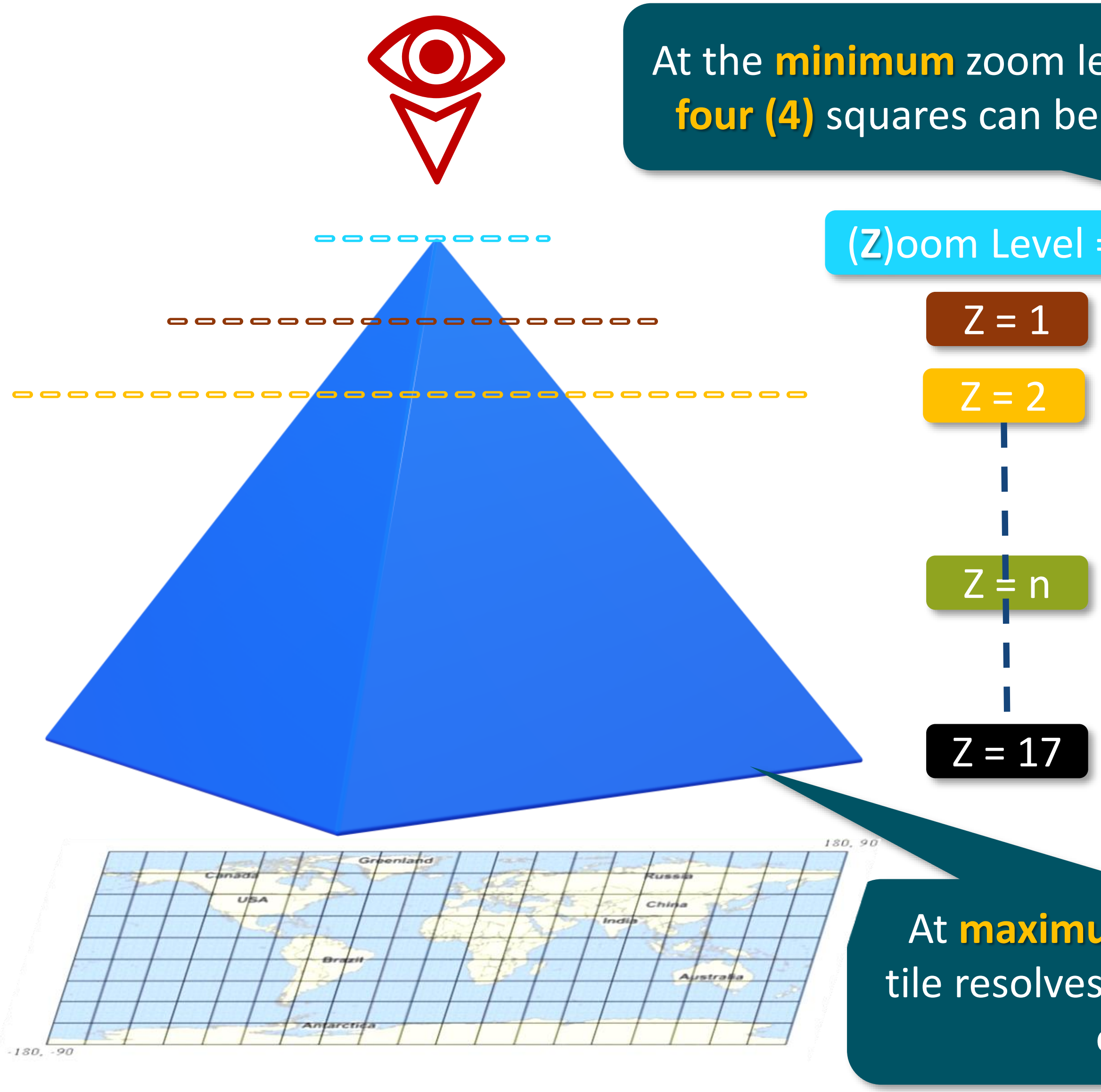


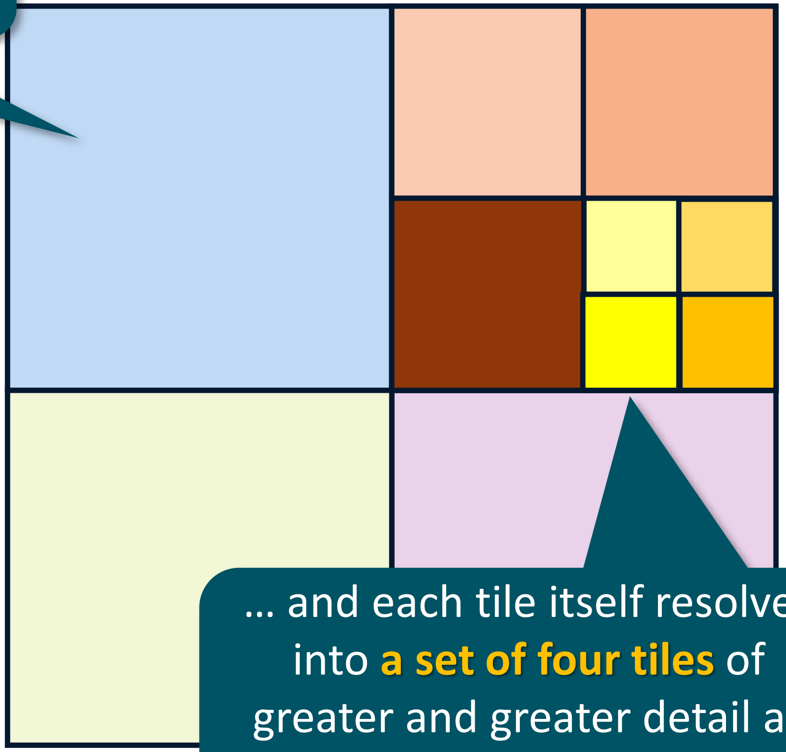
… and plot **distinct points** on a map with just the underlying map background as its **corresponding lowest layer**



We can then reference ***any place*** in the world by its tile's **{X,Y} pair** …

Understanding Vector Tiles: Visualizing a Series of Ever-Smaller Squares





At the **minimum** zoom level, precisely **four (4)** squares can be observed …

… and each tile itself resolves into **a set of four tiles** of greater and greater detail as each zoom level is **traversed**

At **maximum** zoom level, each tile resolves to **just a few meters** on a side!

Handling Vector Tiles: SDO_UTIL Functions and Procedures

Procedure or Function	Purpose
GET_VECTORTILE	Generates a BLOB that represents a **Mapbox Vector Tile** (MVT)
GET_TILE_ENVELOPE	Converts a **tile address** back into a **tile envelope** (i.e. an SDO geometry describing a square)

Yep, it’s really that simple!
Of course, the devil is in the details **here**

GET_VECTORTILE: A Simple Example

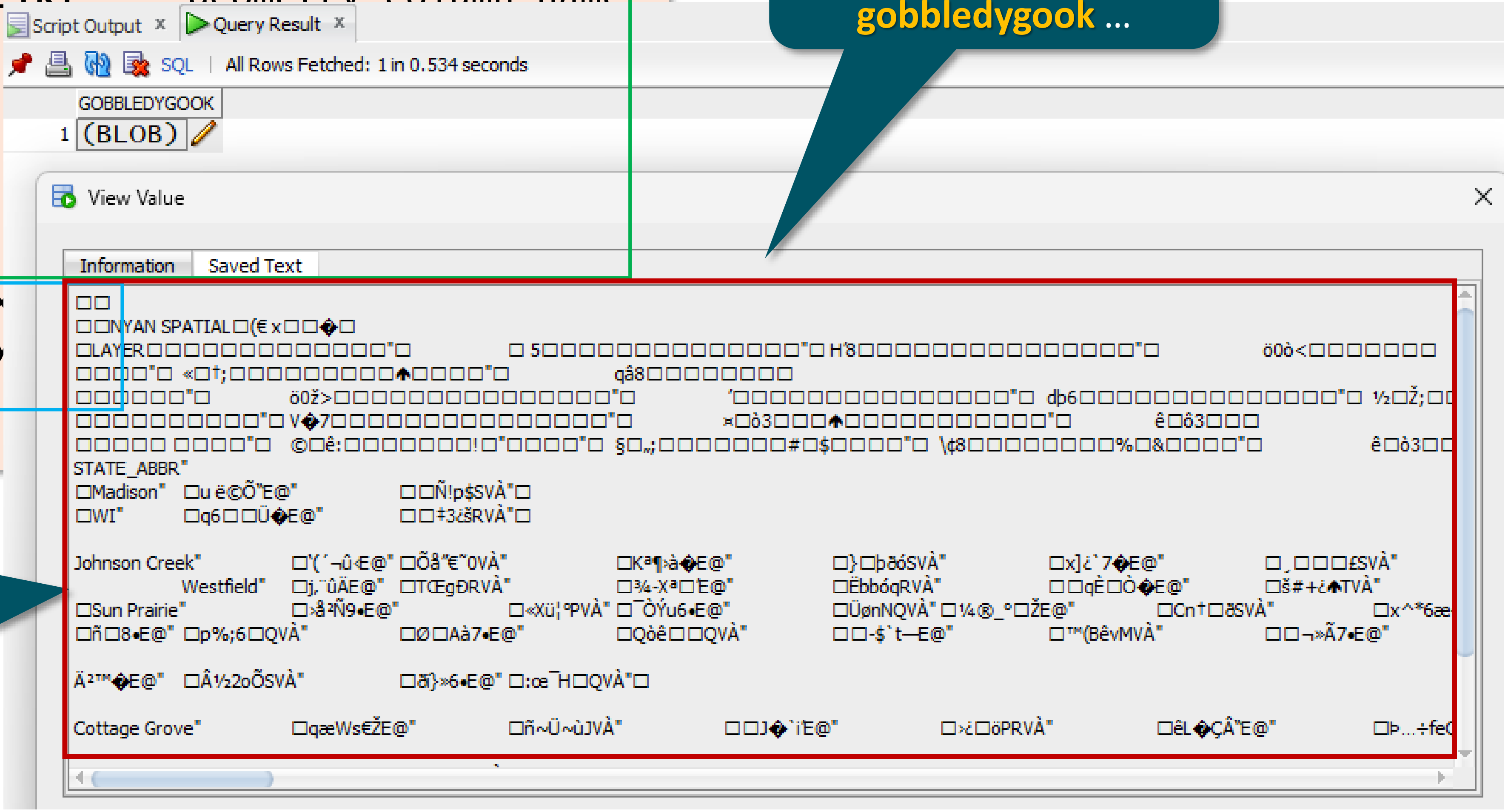
```
SELECT
  SDO_UTIL.GET_VECTORTILE(
    table_name => 'EV_CHARGING_STATIONS’ -- table name
  , geom_col_name => 'GEOMETRY’ -- geometry column name
  , att_col_names =>
    SDO_STRING_ARRAY(
      'CITY’
     ,'STATE_ABBR’
     ,'LONGITUDE’
     ,'LATITUDE’) -- other table attributes
  , tile_x => 129 -- tile x
  , tile_y => 187 -- tile y
  , tile_zoom => 9 -- tile zoom
  ) AS gobbledygook;
```

This query retrieves data from the specified table, its standard columns, and one with a datatype of **SDO_GEOMETRY** …

… but only for this **specific X:Y tile** found at this **specified (Z)oom level**

GET_VECTORTILE: A Simple Example

```
SELECT
  SDO_UTIL.GET_VECTORTILE(
    table_name => 'EV_CHARGING_STATIONS' -- table name
  , geom_col_name => 'GEOMETRY' -- geometry column name
  , att_col_names =>
    SDO_STRING_ARRAY(
      'CITY'
     ,'STATE_ABBR'
     ,'LONGITUDE'
     ,'LATITUDE') -- other
  , tile_x => 129 -- tile x
  , tile_y => 187 -- tile y
  , tile_zoom => 9 -- tile
  ) AS gobbledygook;
```



All Rows Fetched: 1 in 0.534 seconds

	GOBBLEDYGOOK
1	(BLOB)

View Value

```
□□
□□NYAN SPATIAL□(€ x□□�□
□LAYER□□□□□□□□□□□□□□"□        □ 5□□□□□□□□□□□□□□□□"□ H'8□□□□□□□□□□□□□□□□□"□        ö0ò<□□□□□□□□
□□□□"□ «□†;□□□□□□□□□□□□□□"□        qâ8□□□□□□□□□□
□□□□□□"□        ö0ž>□□□□□□□□□□□□□□□□□"□        '□□□□□□□□□□□□□□□□□"□ dþ6□□□□□□□□□□□□□□□"□ ½□Ž;□□
□□□□□□□□□□"□ V�7□□□□□□□□□□□□□□□□□"□        ¤□ò3□□□□□□□□□□□□□□□□□"□        ê□ò3□□□□
□□□□□ □□□□"□ ©□ê:□□□□□□□□!□"□□□□"□ §□„;□□□□□□□□#□$□□□□"□ \¢8□□□□□□□□□%□&□□□□"□        ê□ò3□□
STATE_ABBR"
□Madison"  □u ë©Õ"E@"        □□Ñ!p$SVÀ"□
□WI"       □q6□□Ü�E@"        □□‡3¿šRVÀ"□

Johnson Creek"      □'(´¬û‹E@" □Õå"€˜0VÀ"        □K ª¶›à�E@"        □}□þðóSVÀ"        □x]¿`7�E@"        □¸□□□£SVÀ"
         Westfield"  □j,¨ûÄE@"  □TŒgÐRVÀ"         □¾-Xª□E@"        □ËbbóqRVÀ"        □□qÈ□Ò�E@"        □š#+¿�TVÀ"
□Sun Prairie"        □›å²Ñ9•E@"        □«Xü¦ºPVÀ" □¯ÒYu6•E@"        □ÜønNQVÀ" □¼®_°□ŽE@"        □Cn†□ðSVÀ"        □x^*6æ
□ñ□8•E@"  □p%;6□QVÀ"        □Ø□Aà7•E@"        □QòêDDQVÀ"        □□-$`t—E@"        □™(BêvMVÀ"        □□¬»Ã7•E@"

Ä²™�E@"  □Â½2oÕSVÀ"        □ð}»6•E@" □:œ¯H□QVÀ"□

Cottage Grove"        □qæWs€ŽE@"        □ñ~Ü~ùJVÀ"        □□J�`iE@"        □›¿□öPRVÀ"        □êL�ÇÂ"E@"        □Þ...÷fe
```

To the naked eye, these results appear as total **gobbledygook** ...

... but to a compatible ***mapping tool***, this is a rich source of **points, polygons,** and **descriptors!**

GET_VECTORTILE: Filtering Data Before Tile Creation

```
SELECT
   SDO_UTIL.GET_VECTORTILE(
     table_name => 'EXISTING_PV_SITES'
    ,geom_col_name => 'GEOMETRY'
    ,tile_zoom => 9
    ,tile_x => 129
    ,tile_y => 187
    ,att_col_names => SDO_STRING_ARRAY(
        'USGS_CASE_ID', 'COUNTY', 'STATE_ABBR'
       ,'PV_ARRAY_AREA', 'LONGITUDE', 'LATITUDE')
    ,simple_predicate => SDO_STRING_ARRAY('STATE_ABBR', '=', 'WI')
    ) AS vector_tile
  FROM DUAL;
```

This points to a **single specific tile** at a known zoom level …

… and you can also use simple WHERE clause selection criteria to **limit retrieval of specific data** within that vector tile

Using GET_TILE_ENVELOPE With GET_VECTORTILE

```
SELECT
   SDO_UTIL.GET_VECTORTILE(
     CURSOR(
       WITH cur_tile_envelope(y) AS
         (SELECT SDO_UTIL.GET_TILE_ENVELOPE(tile_zoom => 9, tile_x => 129, tile_y => 187))
       SELECT geometry, usgs_case_id, county, state_abbr, pv_array_area, longitude, latitude
         FROM
           existing_pv_sites
          ,cur_tile_envelope
        WHERE state_abbr = 'WI' AND county IN ('Dane', 'Rock', 'Iowa', 'Grant')
         AND SDO_RELATE(
               geometry, cur_tile_envelope.y
              ,'mask=anyinteract minresolution=' ||
               ((40075016.0 / POWER(2,9)) * 0.001) ||
               ' bypass_point=true') = TRUE)
    ,tile_zoom => 9, tile_x => 129, tile_y => 187, max_features=>256
    ) AS vector_tile
  FROM DUAL;
```

This retrieves **only** data within this single specific tile at its specific zoom level …

… which **SDO_RELATE** can then compare to **any other matching points** within this larger set of data

Note this calculation – it formulates the **minimum size** of the boundary square based on the vector tiles' **zoom factor**

GET_VECTORTILE and Other Spatial Functions: A Final Complex Example

```
WITH g AS (
  SELECT geometry AS g
    FROM existing_pv_sites
   WHERE usgs_case_id = 404985)
SELECT
  SDO_UTIL.GET_VECTORTILE(
    CURSOR(
      WITH x(y) AS (
        SELECT SDO_UTIL.GET_TILE_ENVELOPE(tile_zoom => 9, tile_x => 129, tile_y => 187)
        )
      SELECT
         geometry, usgs_case_id, county, state_abbr, pv_array_area, longitude, latitude
         FROM existing_pv_sites, x, g
        WHERE SDO_WITHIN_DISTANCE(geometry, g, 'distance=' || 2500) = 'TRUE'
          AND SDO_RELATE(
                  geometry, x.y
                 ,'mask=anyinteract minresolution=' ||
                   ((40075016.0 / POWER(2,9)) * 0.001) ||
                 ' bypass_point=true') = TRUE)
       ,tile_zoom => 9, tile_x => 129, tile_y => 187
      ) AS vector_tile
  FROM DUAL;
```

This query compares whether this **single PV array site** …

… exists within a distance of **2500 meters** of the boundary of another larger geographic area

Demo: Leveraging Vector Tiles

Photo by Simon L @ Unsplash

Accessing Vector Tiles Via ORDS REST APIs

```
BEGIN
  ORDS.DEFINE_MODULE(
    P_MODULE_NAME => 'ev_chargers'
  , P_BASE_PATH => '/existing_chargers/'
  , P_ITEMS_PER_PAGE => 50
  , P_STATUS => 'PUBLISHED');
  COMMIT;
END;
/

BEGIN
  ORDS.DEFINE_TEMPLATE(
    P_MODULE_NAME => 'ev_chargers'
  , P_PATTERN => 'vt/:z/:x/:y'
  , P_PRIORITY => 0
  , P_ETAG_TYPE => 'HASH');
  COMMIT;
END;
/
```

First, create the **ORDS REST API module** and **template** …

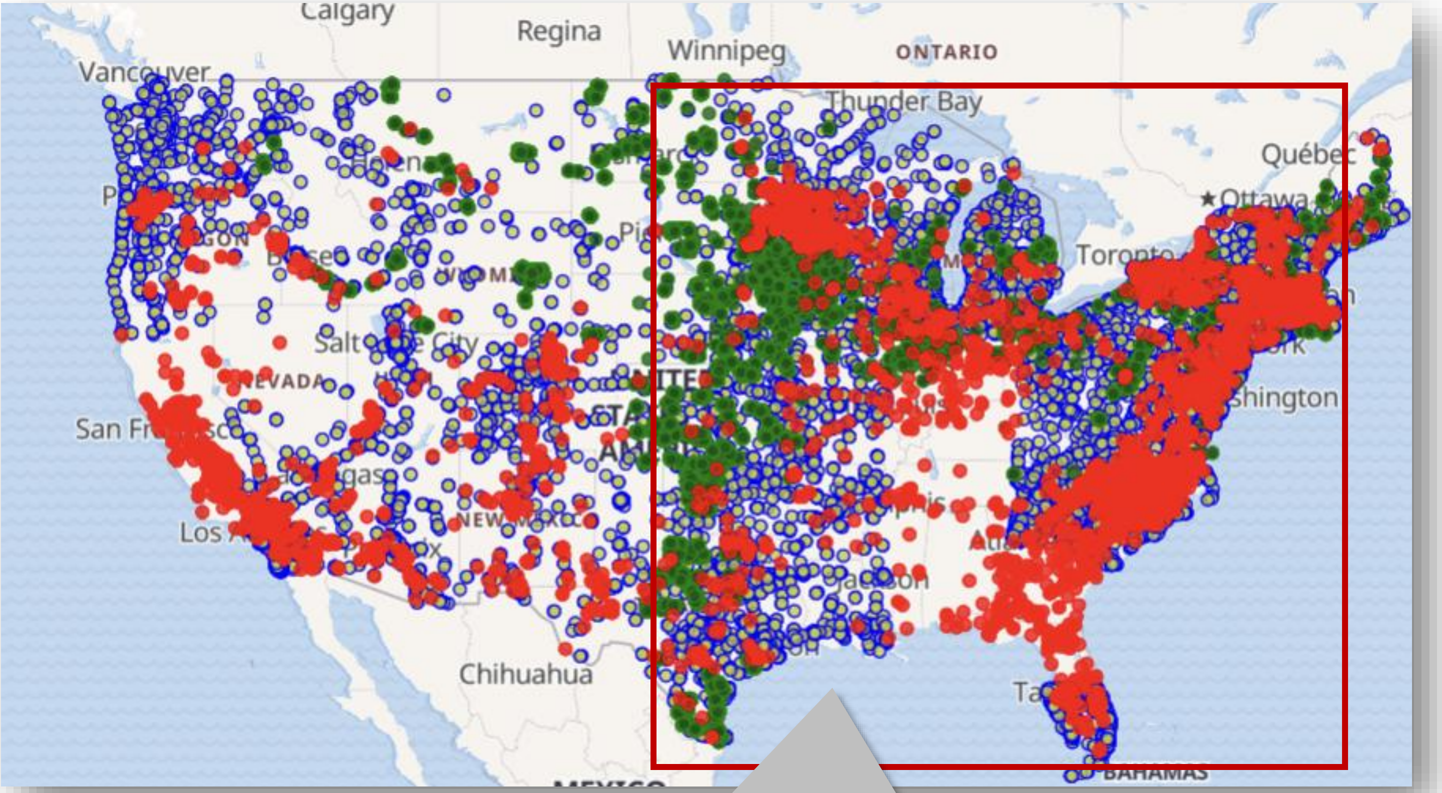
Accessing Vector Tiles Via ORDS REST APIs

```
BEGIN
  ORDS.DEFINE_HANDLER(
    P_MODULE_NAME => 'ev_chargers'
  , P_PATTERN => 'vt/:z/:x/:y'
  , P_METHOD => 'GET'
  , P_SOURCE_TYPE => ORDS.SOURCE_TYPE_MEDIA
  , P_SOURCE => 'SELECT ''application/vnd.mapbox-vector-tile'' AS mediatype
       ,SDO_UTIL.GET_VECTORTILE(
         TABLE_NAME => ''EV_CHARGING_STATIONS''
        ,GEOM_COL_NAME => ''GEOMETRY''
        ,ATT_COL_NAMES => SDO_STRING_ARRAY(''ADDRESS'',''CITY'',''STATE_ABBR‘’
           ,''EV_LEVEL_1'',''EV_LEVEL_2'',''EV_LEVEL_3'',''LATITUDE'',''LONGITUDE'')
        ,TILE_X => :x
        ,TILE_Y_PBF => :y
        ,TILE_ZOOM => :z) AS vtile
      FROM DUAL'
  , P_ITEMS_PER_PAGE => 50);
  COMMIT;
END;
/
```

This tells ORDS to output the result as a defines the output as a BLOB containing **vector tiles** instead of text or other formats

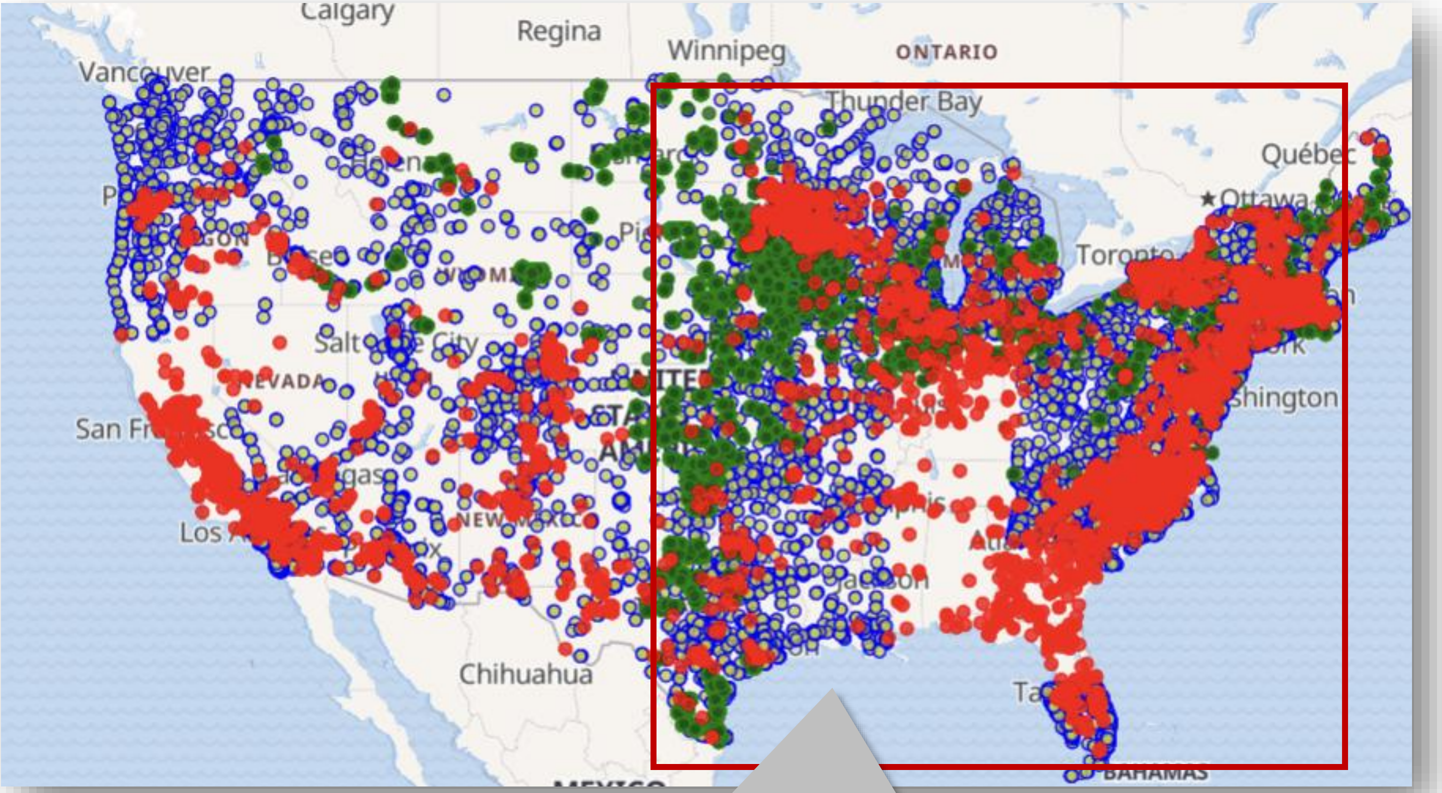
Note we’re mentioning several additional columns here so we can display them in a **pop-up** after a **mouse click**

Displaying Vector Tiles in Multiple Layers



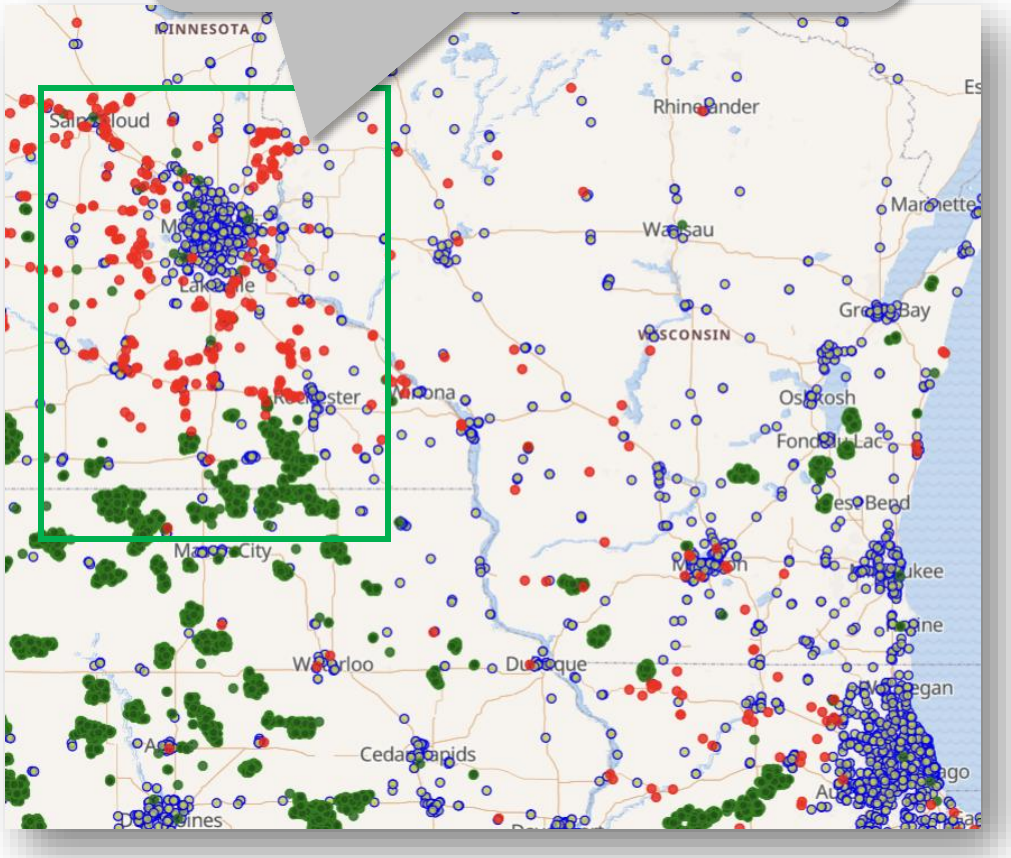
Multiple types of point detail can be layered onto a map – for example, EV chargers, PV arrays, and wind turbines are shown here

Displaying Vector Tiles in Multiple Layers

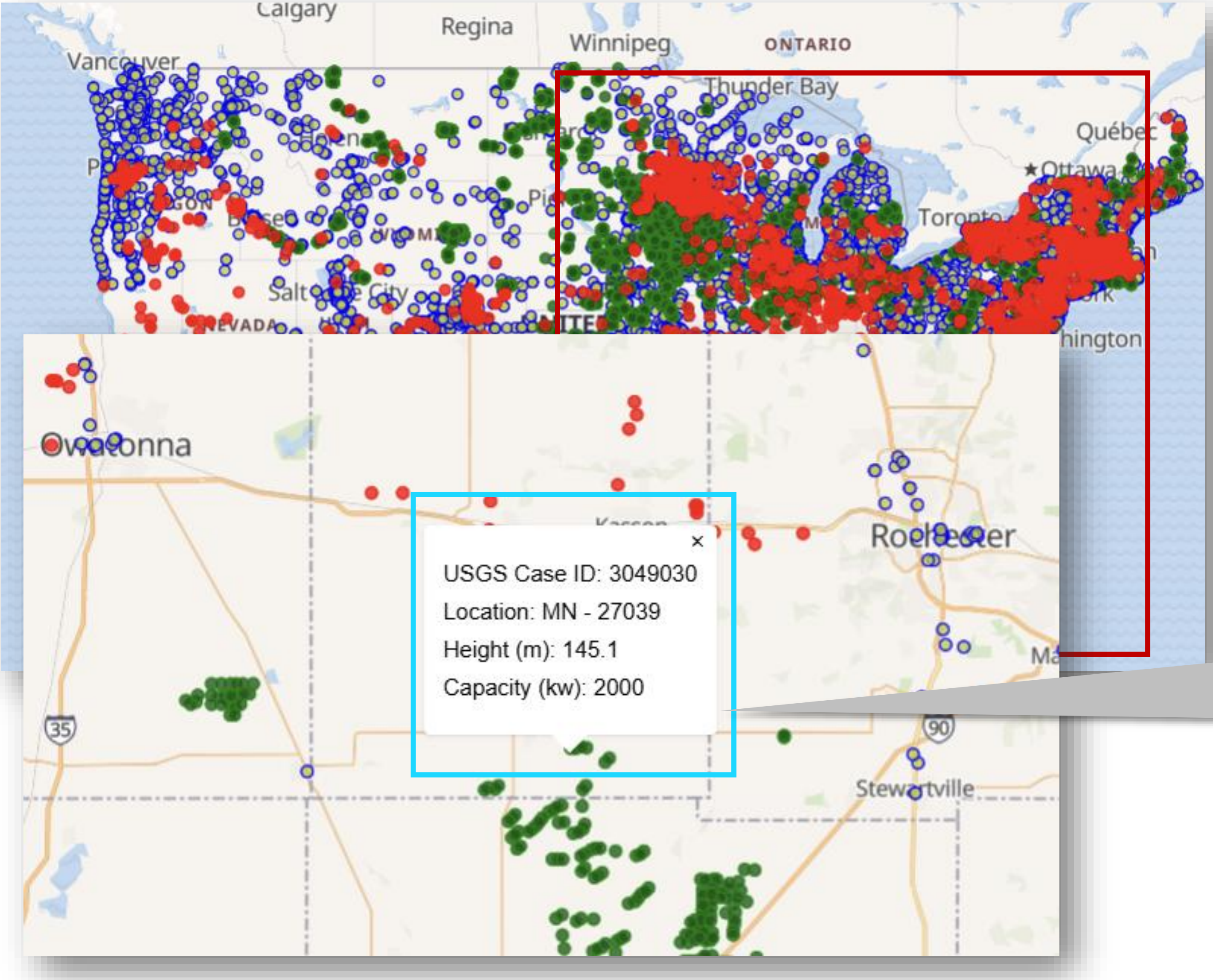


Multiple types of point detail can be layered onto a map – for example, **EV chargers**, **PV arrays**, and **wind turbines** are shown here

We can also drill down to additional detail at lower and lower **zoom levels** to discover additional information at each level …



Displaying Vector Tiles in Multiple Layers



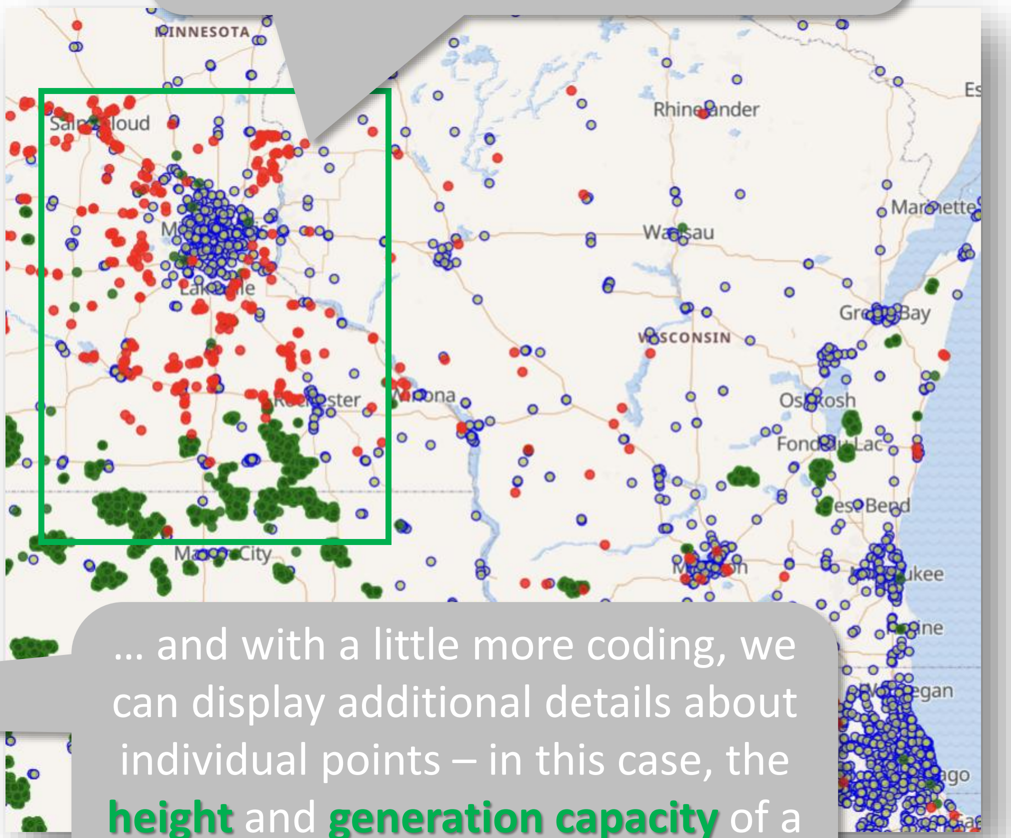
USGS Case ID: 3049030

Location: MN - 27039

Height (m): 145.1

Capacity (kw): 2000

We can also drill down to additional detail at lower and lower zoom levels to discover additional information at each level …



… and with a little more coding, we can display additional details about individual points – in this case, the **height** and **generation capacity** of a single **wind turbine**

Vector Tile Caching

Vector Tile Caching: Saving Frequently-Used Vector Tiles

```
BEGIN
   SDO_UTIL.ENABLE_VECTORTILE_CACHE(
     table_name => 'EXISTING_WIND_TURBINES'
   , geom_col_name => 'GEOMETRY'
   , ts_name => 'DATA'
   , min_zoom => 0
   , max_zoom => 23);
   SDO_UTIL.ENABLE_VECTORTILE_CACHE(
     table_name => 'EXISTING_PV_SITES'
   , geom_col_name => 'GEOMETRY'
   , ts_name => 'DATA'
   , min_zoom => 0
   , max_zoom => 15);
END;
/
```

Oracle 23*ai* also offers the option to **cache vector tiles** for frequent re-use

- Grantable across schemas, should the need arise
- Analogous to the ability to cache **SQL result sets** for queries that gather **a lot** of data that **changes infrequently** (for example, the **master product list** of an automotive company's vehicles)

Vector Tile Caching: Saving Frequently-Used Vector Tiles

```
BEGIN
   SDO_UTIL.ENABLE_VECTORTILE_CACHE(
     table_name => 'EXISTING_WIND_TURBINES'
   , geom_col_name => 'GEOMETRY'
   , ts_name => 'DATA'
   , min_zoom => 0
   , max_zoom => 23);
   SDO_UTIL.ENABLE_VECTORTILE_CACHE(
     table_name => 'EXISTING_PV_SITES'
   , geom_col_name => 'GEOMETRY'
   , ts_name => 'DATA'
   , min_zoom => 0
   , max_zoom => 15);
END;
/
```

This table’s vector tiles will be cached **down to level 23**, the maximum …

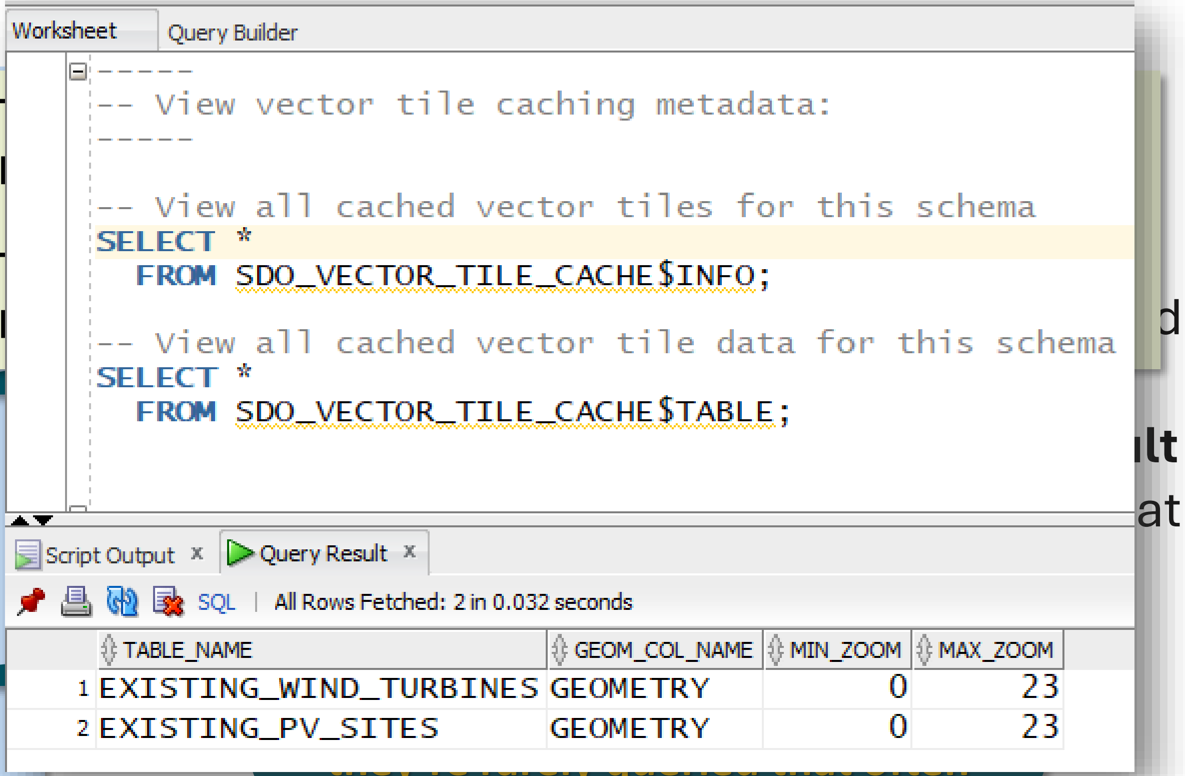
… while this table may have more extensive detail at lower levels, but **they’re rarely queried that often**

Vector Tile Caching: Saving Frequently-Used Vector Tiles

```
BEGIN
                                        CACHE(
                                        RY'

  , max_zoom => 23);
  SDO_UTIL.ENABLE_VECTORTILE_CACHE(
    table_name => 'EXISTING_PV_SITES'
  , geom_col_name => 'GEOMETRY'
  , ts_name => 'DATA'
  , min_zoom => 0
  , max_zoom => 15);
END;
/
```

```
-- View a
SELECT *

-- View a
SELECT *
```

Report which vector tile caches **are in use**, and **what data** they actually contain



```
-----
-- View vector tile caching metadata:
-----

-- View all cached vector tiles for this schema
SELECT *
  FROM SDO_VECTOR_TILE_CACHE$INFO;

-- View all cached vector tile data for this schema
SELECT *
  FROM SDO_VECTOR_TILE_CACHE$TABLE;
```

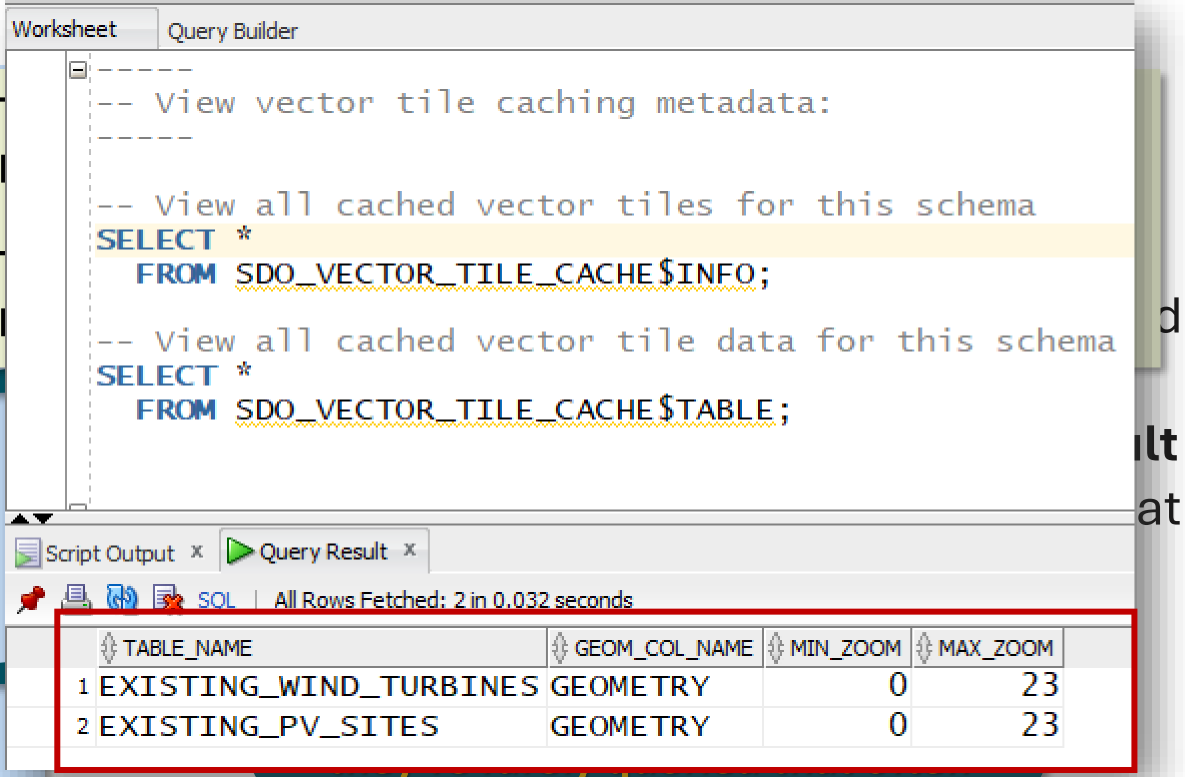
	TABLE_NAME	GEOM_COL_NAME	MIN_ZOOM	MAX_ZOOM
1	EXISTING_WIND_TURBINES	GEOMETRY	0	23
2	EXISTING_PV_SITES	GEOMETRY	0	23

Vector Tile Caching: Saving Frequently-Used Vector Tiles

```
BEGIN
                                            CACHE(
                                          RY'
  , max_zoom => 23);
  SDO_UTIL.ENABLE_VECTORTILE_CACHE(
     table_name => 'EXISTING_PV_SITES'
   , geom_col_name => 'GEOMETRY'
   , ts_name => 'DATA'
   , min_zoom => 0
   , max_zoom => 15);
END;
/
```

Report which vector tile caches **are in use**, and **what data** they actually contain

```
-- View a
SELECT * 

-- View a
SELECT * 
```



```
-----
-- View vector tile caching metadata:
-----

-- View all cached vector tiles for this schema
SELECT *
  FROM SDO_VECTOR_TILE_CACHE$INFO;

-- View all cached vector tile data for this schema
SELECT *
  FROM SDO_VECTOR_TILE_CACHE$TABLE;
```

All Rows Fetched: 2 in 0.032 seconds

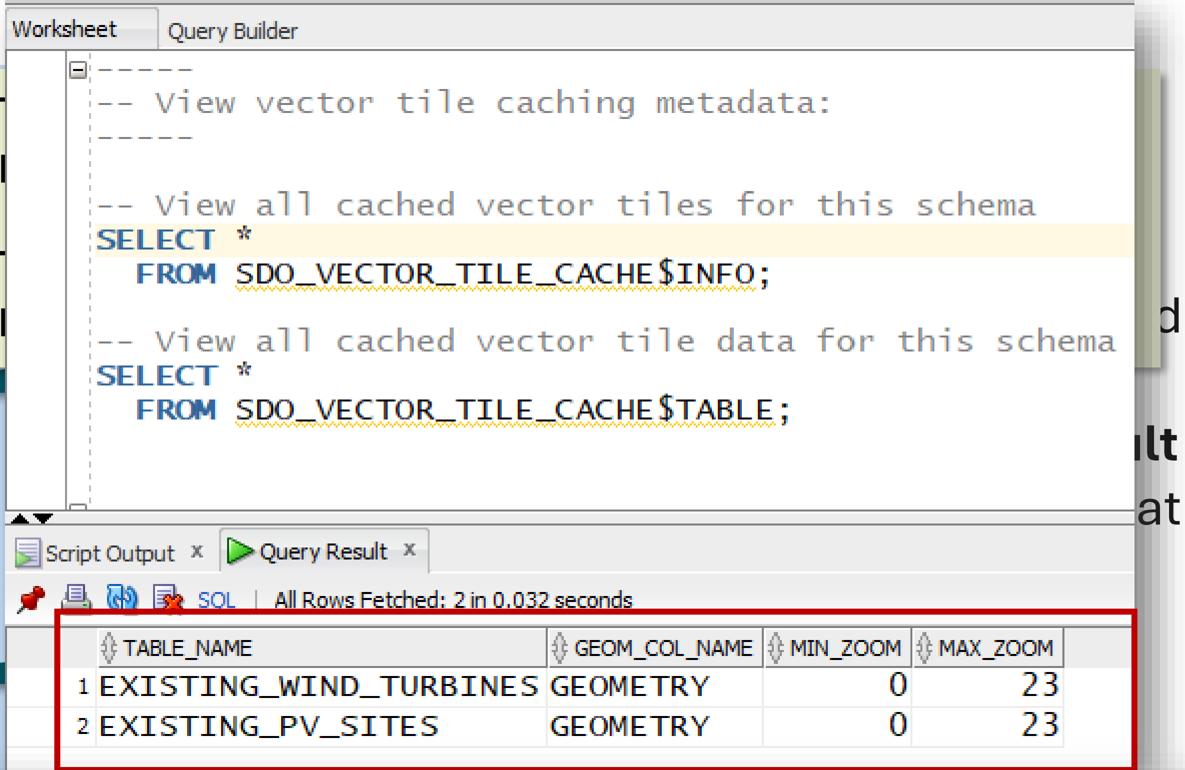
	TABLE_NAME	GEOM_COL_NAME	MIN_ZOOM	MAX_ZOOM
1	EXISTING_WIND_TURBINES	GEOMETRY	0	23
2	EXISTING_PV_SITES	GEOMETRY	0	23

Vector Tile Caching: Saving Frequently-Used Vector Tiles

```
BEGIN
                                        CACHE(
                                         RY'
  , max_zoom => 23);
  SDO_UTIL.ENABLE_VECTORTILE_CACHE(
    table_name => 'EXISTING_PV_SITES'
  , geom_col_name => 'GEOMETRY'
  , ts_name => 'DATA'
  , min_zoom => 0
  , max_zoom => 15);
END;
/
```

Report which vector tile caches **are in use**, and **what data** they actually contain

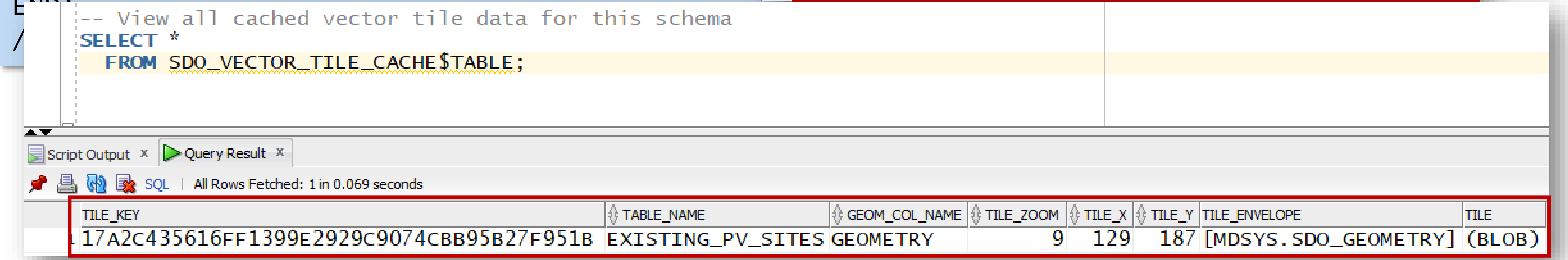
```
-- View a
SELECT * F

-- View a
SELECT * F
```



```
-----
-- View vector tile caching metadata:
-----

-- View all cached vector tiles for this schema
SELECT *
  FROM SDO_VECTOR_TILE_CACHE$INFO;

-- View all cached vector tile data for this schema
SELECT *
  FROM SDO_VECTOR_TILE_CACHE$TABLE;
```

All Rows Fetched: 2 in 0.032 seconds

	TABLE_NAME	GEOM_COL_NAME	MIN_ZOOM	MAX_ZOOM
1	EXISTING_WIND_TURBINES	GEOMETRY	0	23
2	EXISTING_PV_SITES	GEOMETRY	0	23



```
-- View all cached vector tile data for this schema
SELECT *
  FROM SDO_VECTOR_TILE_CACHE$TABLE;
```

All Rows Fetched: 1 in 0.069 seconds

	TILE_KEY	TABLE_NAME	GEOM_COL_NAME	TILE_ZOOM	TILE_X	TILE_Y	TILE_ENVELOPE	TILE
1	17A2C435616FF1399E2929C9074CBB95B27F951B	EXISTING_PV_SITES	GEOMETRY	9	129	187	[MDSYS.SDO_GEOMETRY]	(BLOB)

Vector Tile Caching:Managing Retention and Access

```
BEGIN
   SDO_UTIL.PURGE_VECTORTILE_CACHE(
       table_name => 'EXISTING_WIND_TURBINES’
   );
END;
/

BEGIN
   SDO_UTIL.DISABLE_VECTORTILE_CACHE(
      table_name => 'EXISTING_WIND_TURBINES'
    , geom_col_name => 'GEOMETRY'
   );
END;
/
```

Purge all cached vector tile data for the **EXISTING_WIND_TURBINES** table

Disable vector tile caching for **only** the **GEOMETRY** column in **EXISTING_WIND_TURBINES**

Vector Tile Caching:Managing Retention and Access

```
BEGIN
   SDO_UTIL.PURGE_VECTORTILE_CACHE(
      table_name => 'EXISTING_WIND_TURBINES'
   );
END;
/

BEGIN
                                                S'
E
/
```

```
BEGIN
   SDO_UTIL.GRANT_VECTORTILE_CACHE(
     schema_name => 'GEOSWARM'
   , read_only => TRUE
   );
END;
/

BEGIN
   SDO_UTIL.REVOKE_VECTORTILE_CACHE(
     schema_name => 'HOL23'
   );
END;
/
```

Purge all cached vector tile data for the **EXISTING_WIND_TURBINES** table

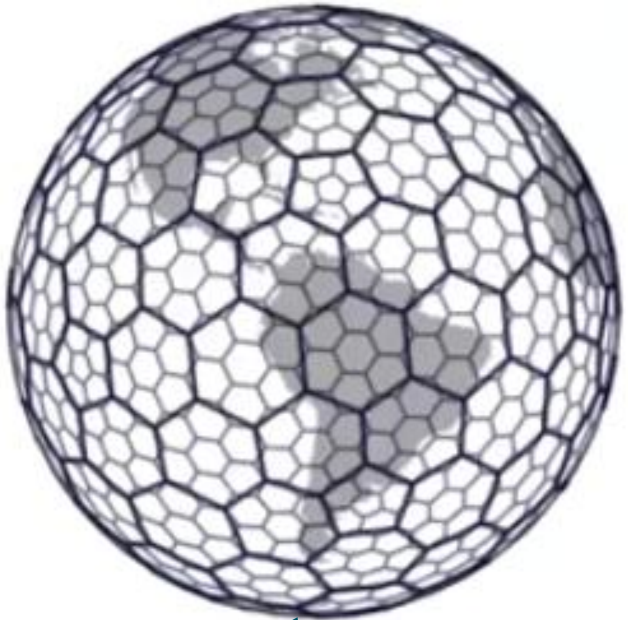
Disable vector tile caching for **only** the **GEOMETRY** column in **EXISTING_WIND_TURBINES**

Grant a different schema to access all cached vector tile data in the current schema

Revoke vector tile caching capabilities for a specific schema

Hexagonal Hierarchical (H3) Spatial Indexing

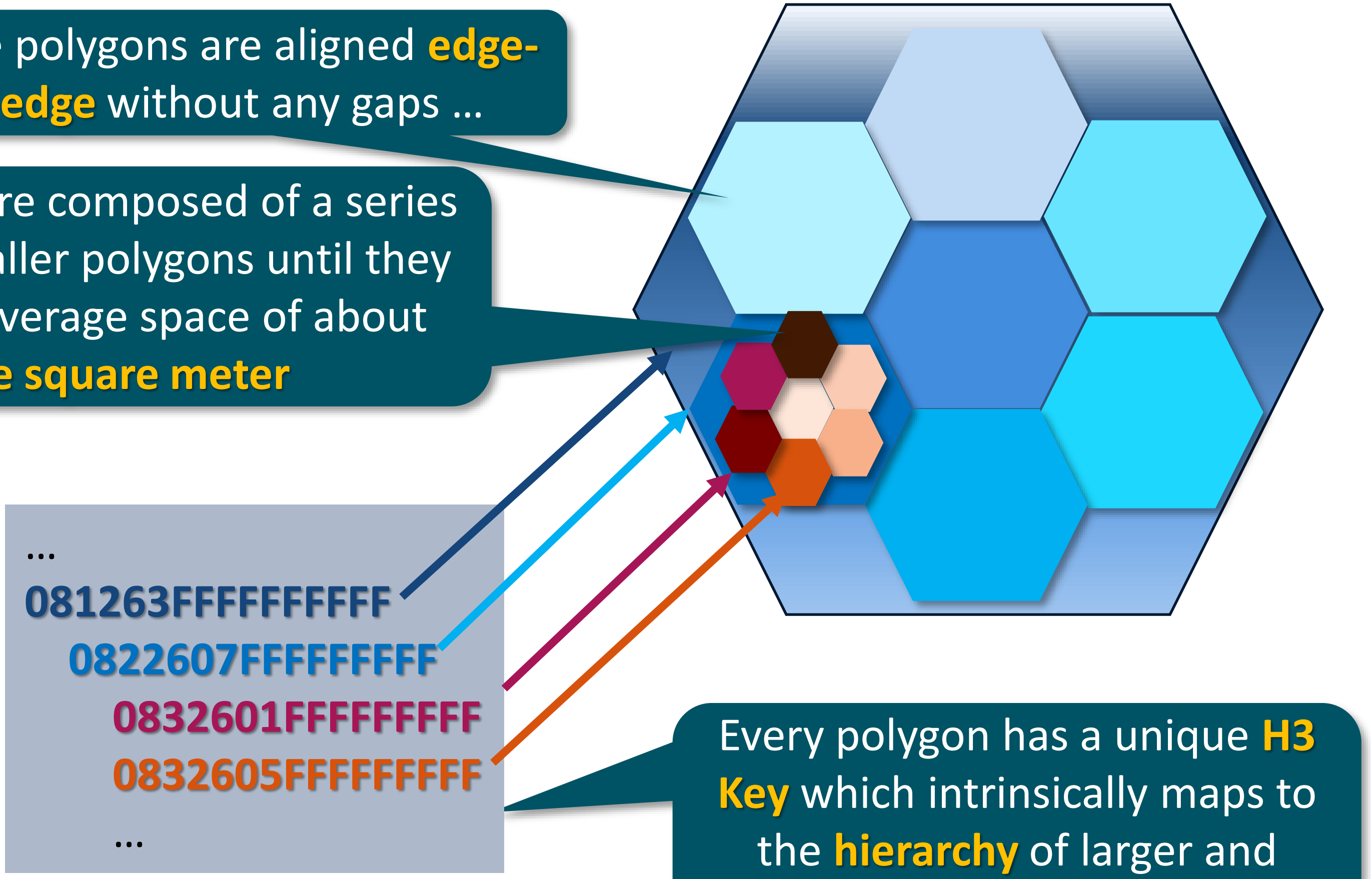
H3 Spatial Indexing: Why Our Uber Driver **Always** Finds Us.



Back in November 2018, engineers at Uber built a new **geographic indexing system** that covers every square meter of our planet as a series of 110 **hexagons** (and 12 **pentagons**)

These polygons are aligned **edge-to-edge** without any gaps …

… and they're composed of a series of ever-smaller polygons until they reach a coverage space of about **one square meter**



…
081263FFFFFFFFF
0822607FFFFFFFF
0832601FFFFFFFF
0832605FFFFFFFF
…

Every polygon has a unique **H3 Key** which intrinsically maps to the **hierarchy** of larger and smaller polygons

Exhaustive detailed information on the H3 standard is **here**

H3 SDO_UTIL Functions & Procedures: The Basics

Procedure or Function	Purpose
H3_KEY	Computes **H3 cell** for a given geodetic point and returns requested **statistic** for the given H3 level
H3_MEMBER	Returns the **MBR** of the given H3 **cell**
H3_PARENT	Computes **containing H3 cell** (64-bit (8 byte RAW)) for given **cell** based on the H3 **hierarchy**
H3SUM_CREATE_TABLE	**Creates** an H3 summary table for geodetic point data
H3SUM_AS_TABLE	**Returns rows** from an H3 summary table as table of type **H3SUM_TAB_T**
H3SUM_GET_CURSOR	Creates a **cursor** for results of searching an H3 summary table with specified level number and search MBR
H3SUM_VECTORTILE	Creates and returns specified **MVT vector tile** from an H3 summary table created with **H3SUM_CREATE_TABLE**

Combining Disparate GIS Datasets Based On H3 KEY Values

```
SELECT h3key
, COUNT(pv_case_id) AS num_h3_locations
, SUM(facility_capacity_ac) AS tot_pv_ac_cap
, SUM(facility_capacity_dc) AS tot_pv_dc_cap
, SUM(turbine_capacity) AS tot_wt_cap
  FROM (
    WITH pv AS (
      SELECT usgs_case_id AS pv_case_id
        , state_abbr
        , county
        , pv_array_area
        , facility_capacity_ac
        , facility_capacity_dc
        , SDO_UTIL.H3_KEY(longitude, latitude,
          FROM existing_pv_sites)
 ,wt AS (
      SELECT usgs_case_id AS wt_case_id
        , state_abbr
        , county
        , turbine_capacity
        , SDO_UTIL.H3_KEY(geometry, 5) AS h3key
          FROM existing_wind_turbines)
. . .
```

```
. . .
SELECT pv.h3key
      , pv.pv_case_id
      , pv.state_abbr
      , pv.county
      , pv.pv_array_area
      , pv.facility_capacity_ac
      , pv.facility_capacity_dc
      , wt.wt_case_id
      , NVL(wt.turbine_capacity,0) AS turbine_capacity
   FROM pv, wt
  WHERE pv.h3key = wt.h3key
  ORDER BY pv.state_abbr, pv.county)
GROUP BY h3key;
```

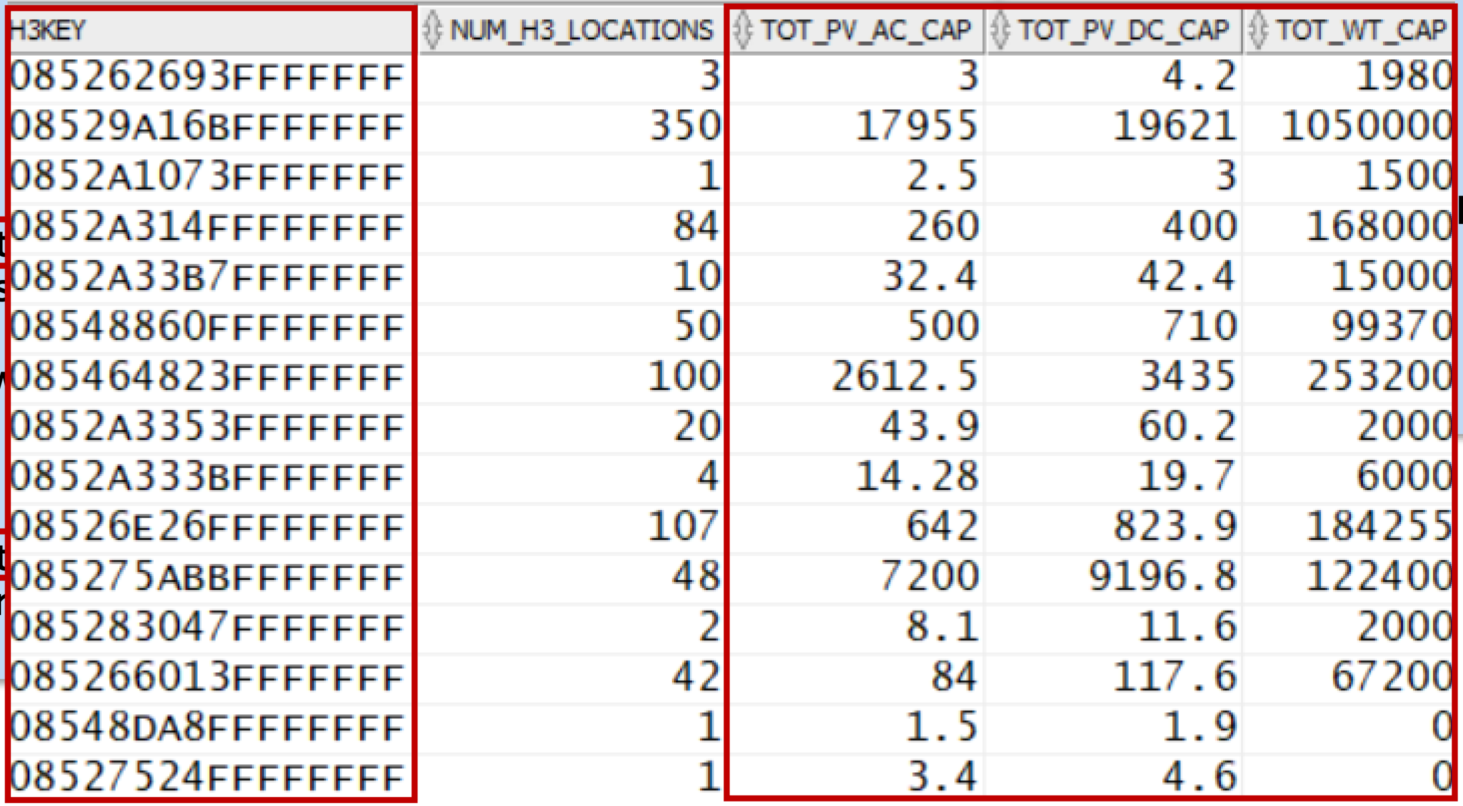
… so these two areas can be **joined together at that same resolution** to see what overlaps occur in those geographies

Combining Disparate GIS Datasets Based On H3 KEY Values

```
SELECT h3key
, COUNT(pv_case_id) AS num_h3_locations
, SUM(facility_capacity_ac) AS tot_pv_ac_cap
, SUM(facility_capacity_dc) AS tot_pv_dc_cap
, SUM(turbine_capacity) AS tot_wt_cap
  FROM (
    WITH pv AS (
      SELECT usgs_case_id AS pv_case_id
      , state_abbr
      , county
      , pv_array_area
      , facility_capacity_ac
      , facility_capacity_dc
      , SDO_UTIL.H3_KEY(longit
        FROM existing_pv_sites
 ,wt AS (
      SELECT usgs_case_id AS w
      , state_abbr
      , county
      , turbine_capacity
      , SDO_UTIL.H3_KEY(geomet
        FROM existing_wind_tur
. . .
```

```
. . .
SELECT pv.h3key
      , pv.pv_case_id
      , pv.state_abbr
      , pv.county
                                    pacity
```

There is indeed at least some overlap between those two areas; we can calculate the **combined power capacities** for both disjoint data sets



H3KEY	NUM_H3_LOCATIONS	TOT_PV_AC_CAP	TOT_PV_DC_CAP	TOT_WT_CAP
085262693FFFFFFF	3	3	4.2	1980
08529A16BFFFFFFF	350	17955	19621	1050000
0852A1073FFFFFFF	1	2.5	3	1500
0852A314FFFFFFFF	84	260	400	168000
0852A33B7FFFFFFF	10	32.4	42.4	15000
08548860FFFFFFFF	50	500	710	99370
085464823FFFFFFF	100	2612.5	3435	253200
0852A3353FFFFFFF	20	43.9	60.2	2000
0852A333BFFFFFFF	4	14.28	19.7	6000
08526E26FFFFFFFF	107	642	823.9	184255
085275ABBFFFFFFF	48	7200	9196.8	122400
085283047FFFFFFF	2	8.1	11.6	2000
085266013FFFFFFF	42	84	117.6	67200
08548DA8FFFFFFFF	1	1.5	1.9	0
08527524FFFFFFFF	1	3.4	4.6	0

H3 SDO_UTIL Functions & Procedures: Cell Boundaries, Areas, & Accuracy

Procedure or Function	Purpose
H3_BASE_CELL	Returns H3 **base** cell (64-bit, 8 byte RAW) matching given H3 cell
H3_BOUNDARY	Computes a **polygon** representing the given cell
H3_CENTER	Computes center of the given H3 cell
H3_PENTAGON_AREA	Returns **size** of a pentagon at given resolution
H3_PENTAGON_EDGELEN	Returns **length of an edge** of a pentagon at given resolution
H3_RESOLUTION	Returns **resolution** of an H3 cell from **0** (coarsest) to **15** (finest)
H3SUM_ESTIMATE_RESOLUTION	Provides an H3 resolution returning ***approximately*** requested number of hexes in given tile

H3 SDO_UTIL Functions & Procedures: Cell-Level

Procedure or Function	Purpose
H3_NUM_CELLS	Returns **number of H3 cells** covering the Earth at the specified **resolution**
H3_HEX_AREA	Returns **minimum**, **maximum** or **average area** of hexes for given H3 level
H3_HEX_EDGELEN	Returns **minimum**, **maximum** or **average edge length** (EDGELEN) of hexes at the given resolution
H3_IS_CLASS3	Returns **BOOLEAN** value that determines if the given cells at the given resolution are “Class 3” in Uber’s H3 system
H3_IS_PENTAGON	Returns **BOOLEAN** value that determines whether a given cell is a **pentagon** (instead of a **hexagon**) or not
H3_IS_VALID_CELL	Returns **TRUE** if the given H3 key is **correctly formed** and identifies an H3 cell; otherwise, the procedure returns **FALSE**

Demo: Vector Tiles and H3 Indexing In Combination

Photo by Simon L @ Unsplash

H3 Indexes: Creating a Summary Table

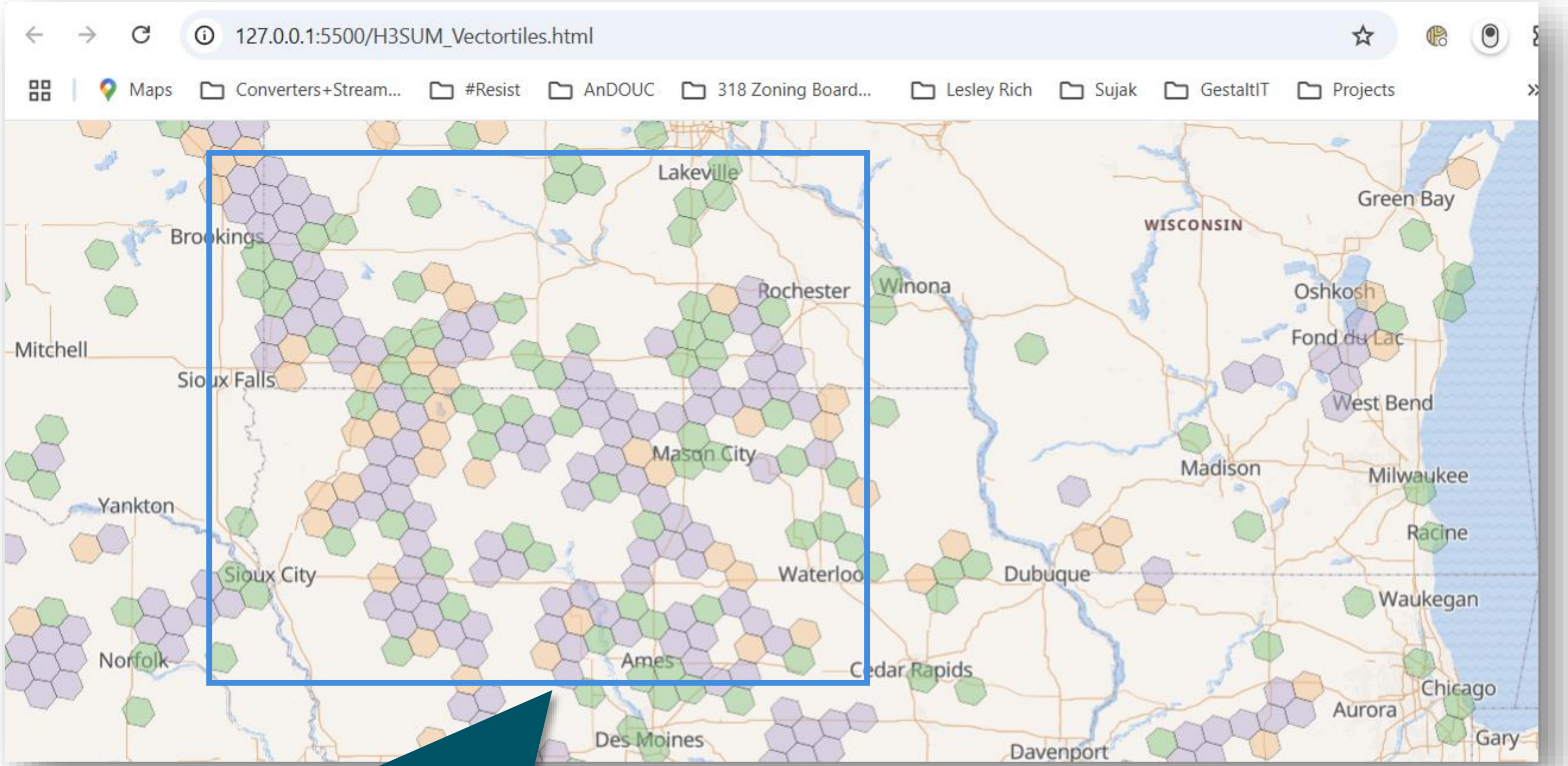
```
DROP TABLE IF EXISTS h3sum_wind_turbines;

DELETE FROM user_sdo_geom_metadata
 WHERE table_name = 'H3SUM_WIND_TURBINES';

BEGIN
  SDO_UTIL.H3SUM_CREATE_TABLE(
    table_out     => 'H3SUM_WIND_TURBINES'
  , table_in      => 'EXISTING_WIND_TURBINES'
  , geomcol_spec => 'GEOMETRY'
  , col_spec      => '1,CNT;TURBINE_CAPACITY,MIN;TURBINE_CAPACITY,MAX; TURBINE_CAPACITY,AVG'
  , max_h3_level => 7
  );

END;
/
```

This describes which tables to use as **source and target** of the H3 summary table operation …

… and this specifies which column(s) should be **aggregated per function** (e.g. MIN, MAX, AVG) or simply counted (CNT)

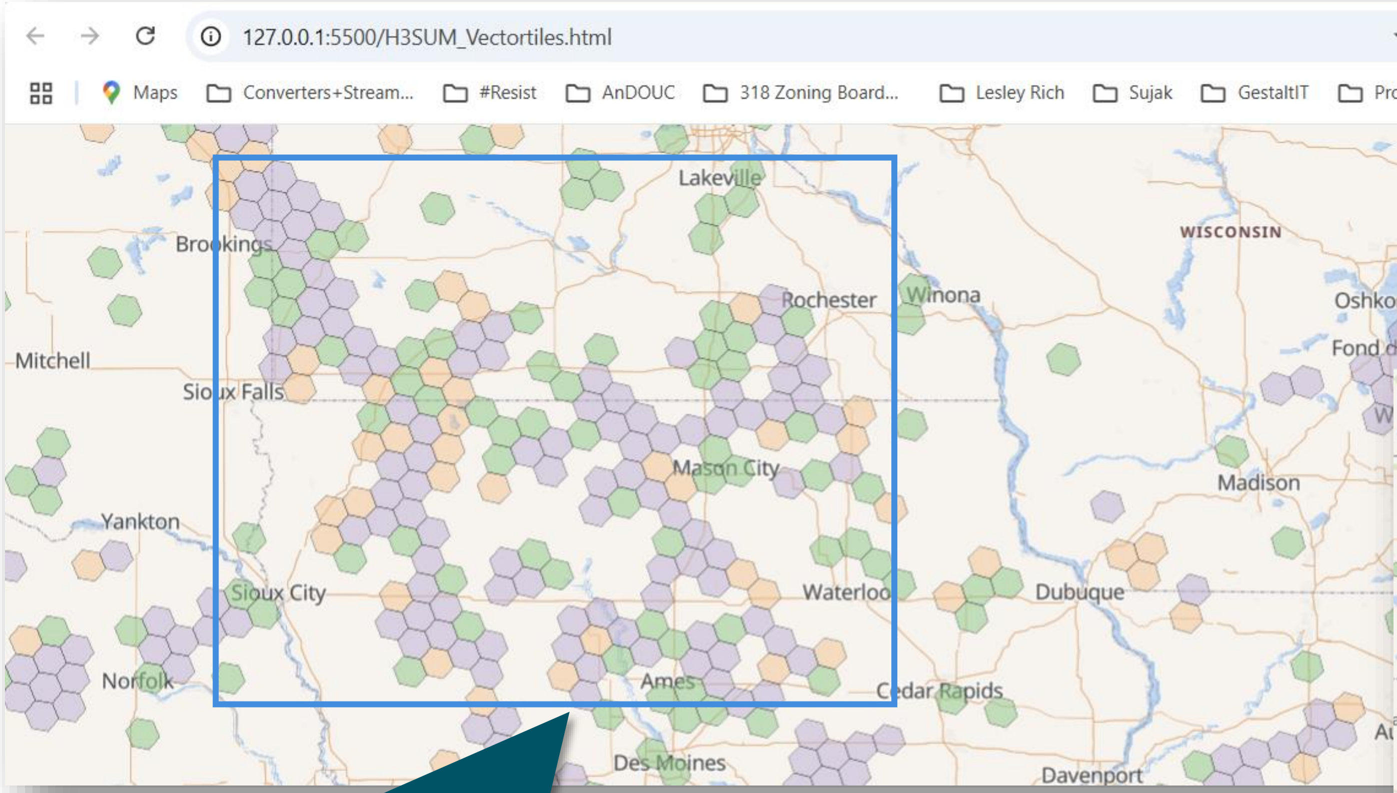
Since there are **16 levels** of H3 key hierarchy, we can set a reasonable limit for the **total number of levels** at which data should be aggregated

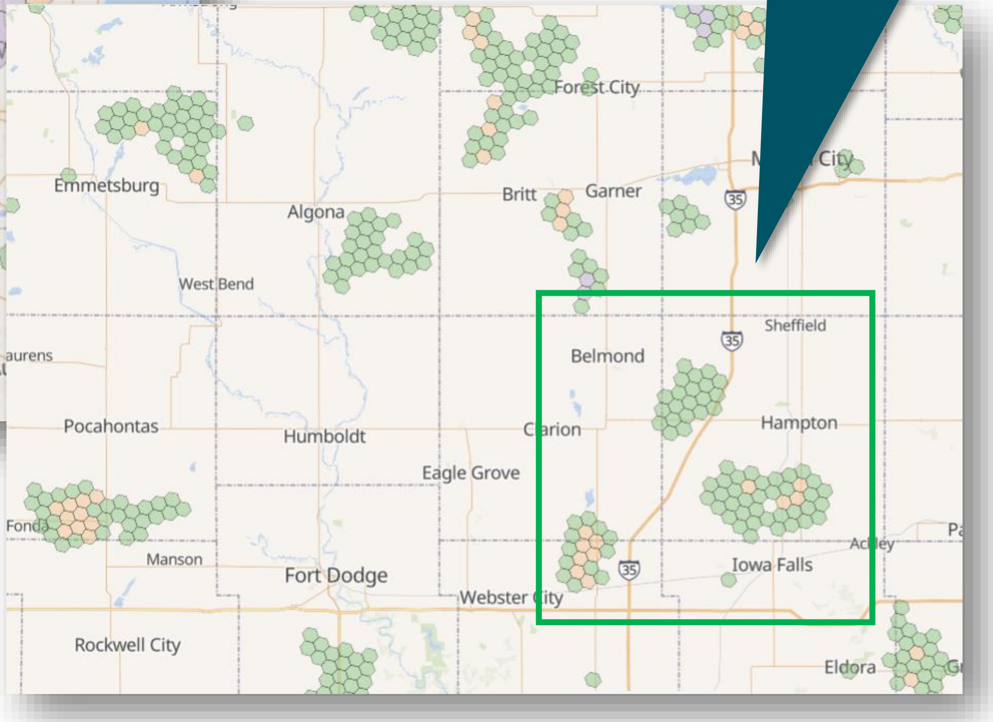
Displaying Vector Tiles Leveraging H3 Summary Information



Each set of H3 summary tiles are **color-coded** to highlight which ones have the **most** wind turbines within their boundaries

Displaying Vector Tiles Leveraging H3 Summary Information



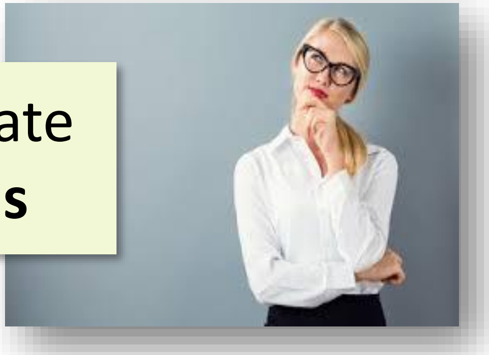


We can also drill down to additional detail at lower and lower **zoom levels** to discover additional information at each level

Each set of H3 summary tiles are **color-coded** to highlight which ones have the **most** wind turbines within their boundaries

Vector Tiling and H3 Indexing: Lessons Learned

Vector tiles offer extremely fast display of **huge** numbers of disparate points in **myriad mapping layers** and **across different zoom levels**



Likewise, **H3 Indexing** offers the ability to display aggregate data **extremely efficiently** within modern GIS systems

These features are accessible through **Oracle Spatial Studio** and **MapBox HTML JavaScript** … and may be coming soon to a **future Oracle APEX release**

Oracle Live Labs, Blog Posts, and Articles on New Spatial Features

Video: Add Fast and Scalable Maps to your Apps with Vector Tiles and H3 in Oracle Database 23ai
https://youtu.be/cQOcNAwHHDc?si=kBJxU8D4VF2VsJ1y

Blog: Add Fast and Scalable Maps to your Apps with Vector Tiles and H3 in Oracle Database 23ai
https://blogs.oracle.com/database/post/make-better-maps-for-your-apps-with-spatial-vector-tiles-and-h3-in-oracle-database-23ai

Video: AskTom Session on OAC, Vector Tile Caching, and SELECT AI:
https://www.youtube.com/watch?v=bKFfTfNUx2A&ab_channel=OracleDevelopers

H3 Indexing:
https://docs.oracle.com/en/database/oracle/oracle-database/23/spatl/h3-indexing.html

Vector Tiles & Vector Tile Cache:
https://docs.oracle.com/en/database/oracle/oracle-database/23/spatl/vector-tiles.html#GUID-77640B2D-C3B2-438F-A46A-0AAD78DB7AB9

Useful Resources, Documentation, and Technical Details

Oracle Spatial & Graph Developers Guide:
https://docs.oracle.com/en/database/oracle/oracle-database/23/vsiad/aivs_genarch.html

Mapbox GL JavaScript Guides:
https://docs.mapbox.com/mapbox-gl-js/guides/

Mapbox GL JavaScript Feature Examples:
https://docs.mapbox.com/mapbox-gl-js/example/

H3: Uber's Hexagonal Hierarchical Spatial Index:
https://www.uber.com/blog/h3/

H3-js - Hexagons in the Browser Video:
https://www.youtube.com/watch?v=BsMIrBHLfLE

Other Interesting Reading

California now has 48% more EV chargers than gasoline nozzles in the state
https://www.gov.ca.gov/2025/03/20/california-now-has-48-more-ev-chargers-than-gasoline-nozzles-in-the-state/

Wind and Solar Overtake Coal Power for First Time in U.S.
https://www.wsj.com/articles/wind-and-solar-overtake-coal-power-for-the-first-time-in-the-u-s-a52e9d8f

Solar Farms Have a Superpower Beyond Clean Energy
https://www.nytimes.com/2024/09/05/climate/solar-power-pollinators-wildlife.html

Promoters of clean-energy data centers in Virginia coal country unfazed by doubters
https://energynews.us/2024/09/10/promoters-of-clean-energy-data-centers-in-virginia-coal-country-unfazed-by-doubters/

Study: EV charging stations boost spending at nearby businesses
https://news.mit.edu/2024/study-ev-charging-stations-boost-nearby-business-spending-0904

IRA credits and energy demand continue to drive renewables investments
https://www.utilitydive.com/news/inflation-reduction-act-credits-energy-demand-investment-financing-trump/742485

Helpful Links –

ORACLE ANALYTICS VIDEOS:
https://www.youtube.com/@OracleAnalytics/videos

OAC SEPTEMBER NEW FEATURES VIDEOS BY ORACLE:
https://bit.ly/OACSept24Features

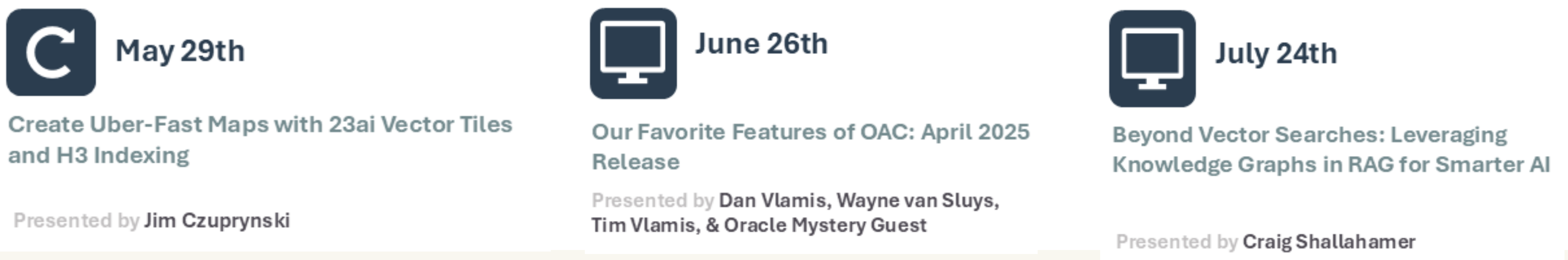
OAC NEW FEATURES DOCUMENTATION BY ORACLE:
https://docs.oracle.com/en/cloud/paas/analytics-cloud/acswn/index.html#GUID-CFF90F44-BCEB-49EE-B40B-8D040F02D476

ORACLE ANALYTICS COMMUNITY:
https://community.oracle.com/products/oracleanalytics

ORACLE ANALYTICS LIBRARY/EXAMPLES:
https://www.oracle.com/business-analytics/data-visualization/examples/

ORACLE ANALYTICS LIVE DEMOS:
https://www.oracle.com/business-analytics/data-visualization/demos/

Future & Past TechCasts:



May 29th

Create Uber-Fast Maps with 23ai Vector Tiles and H3 Indexing

Presented by **Jim Czuprynski**

June 26th

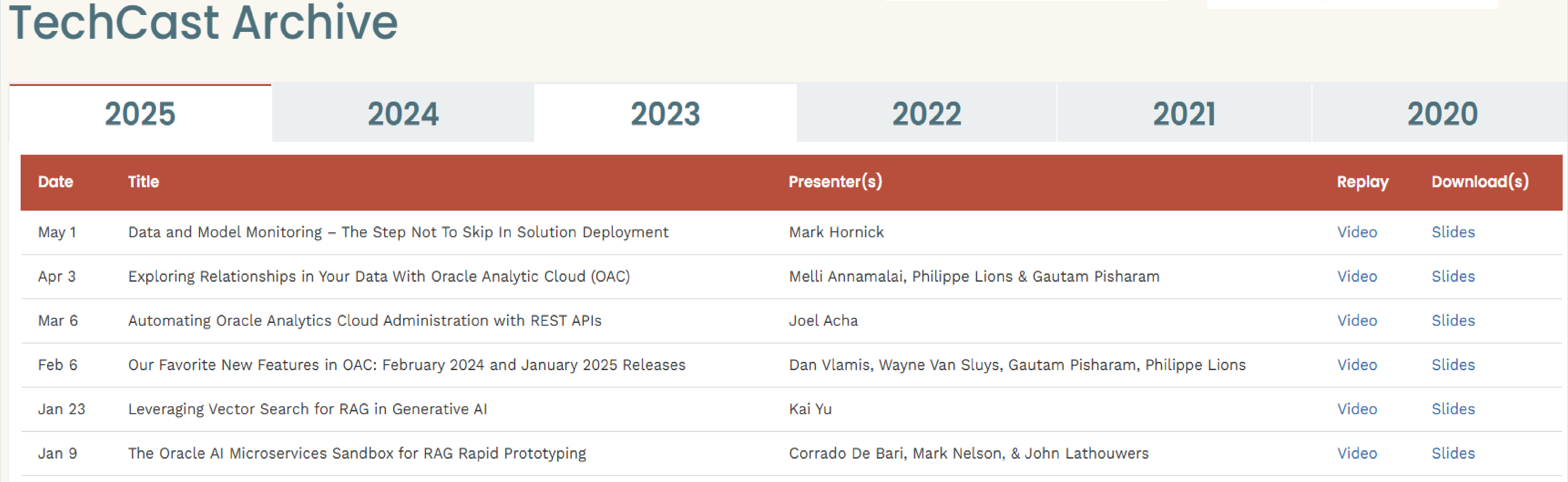
Our Favorite Features of OAC: April 2025 Release

Presented by **Dan Vlamis, Wayne van Sluys, Tim Vlamis, & Oracle Mystery Guest**

July 24th

Beyond Vector Searches: Leveraging Knowledge Graphs in RAG for Smarter AI

Presented by **Craig Shallahamer**



TechCast Archive

2025 2024 2023 2022 2021 2020

Date	Title	Presenter(s)	Replay	Download(s)
May 1	Data and Model Monitoring – The Step Not To Skip In Solution Deployment	Mark Hornick	Video	Slides
Apr 3	Exploring Relationships in Your Data With Oracle Analytic Cloud (OAC)	Melli Annamalai, Philippe Lions & Gautam Pisharam	Video	Slides
Mar 6	Automating Oracle Analytics Cloud Administration with REST APIs	Joel Acha	Video	Slides
Feb 6	Our Favorite New Features in OAC: February 2024 and January 2025 Releases	Dan Vlamis, Wayne Van Sluys, Gautam Pisharam, Philippe Lions	Video	Slides
Jan 23	Leveraging Vector Search for RAG in Generative AI	Kai Yu	Video	Slides
Jan 9	The Oracle AI Microservices Sandbox for RAG Rapid Prototyping	Corrado De Bari, Mark Nelson, & John Lathouwers	Video	Slides

Submit a topic to share at **https://andouc.org/techcasts/**