BIODIVERSITY BUILDING BLOCKS FOR POLICY

Species Occurrence Cubes

Andrew Rodrigues

GBIF

Contributions from Pieter Huybrechts -Research Institute for Nature and Forest, Belgium

Funded by the European Union

GBIF

Transforming biodiversity observations into insights

20/02/2026

SQL Downloads

QUIZ

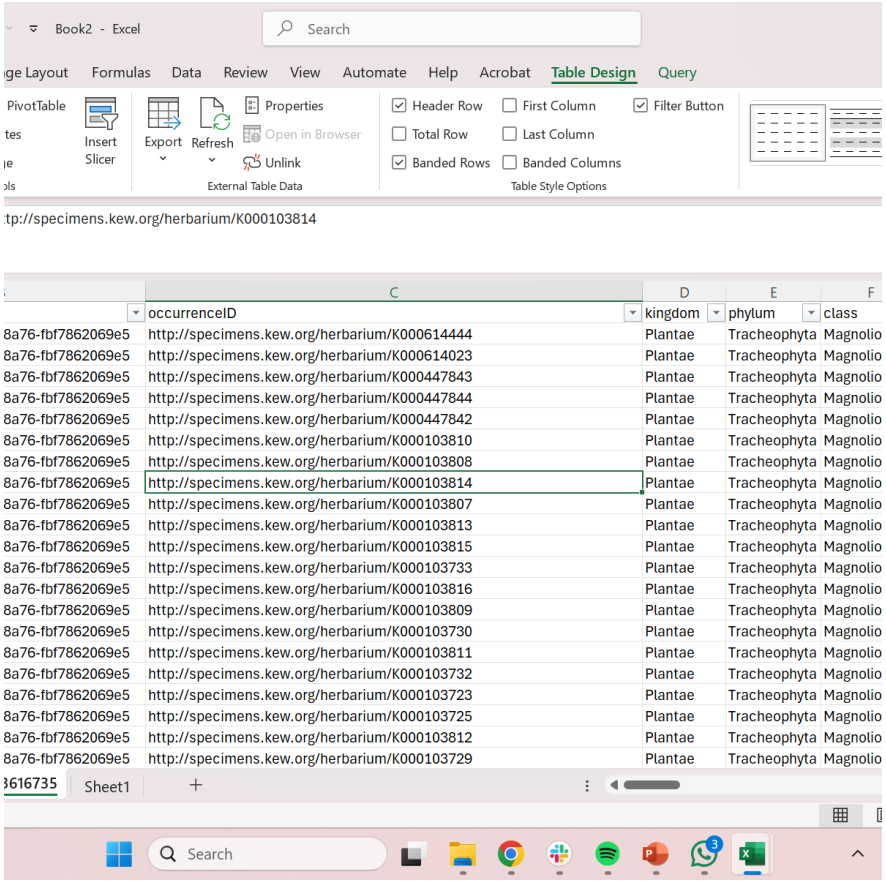
HOW WELL DO YOU KNOW YOUR ISO 3166-1 alpha-2 COUNTRY CODES?

1. Choose one of 3 options for the ISO alpha-2 code for each of the 6 countries in the poll questions.
2. You have 5 seconds to select your option
3. All answers are anonymous
4. If you get all 6, congratulations, you are country code master!

(This does have something to do with the training….just in case you were wondering)

GBIF Downloads

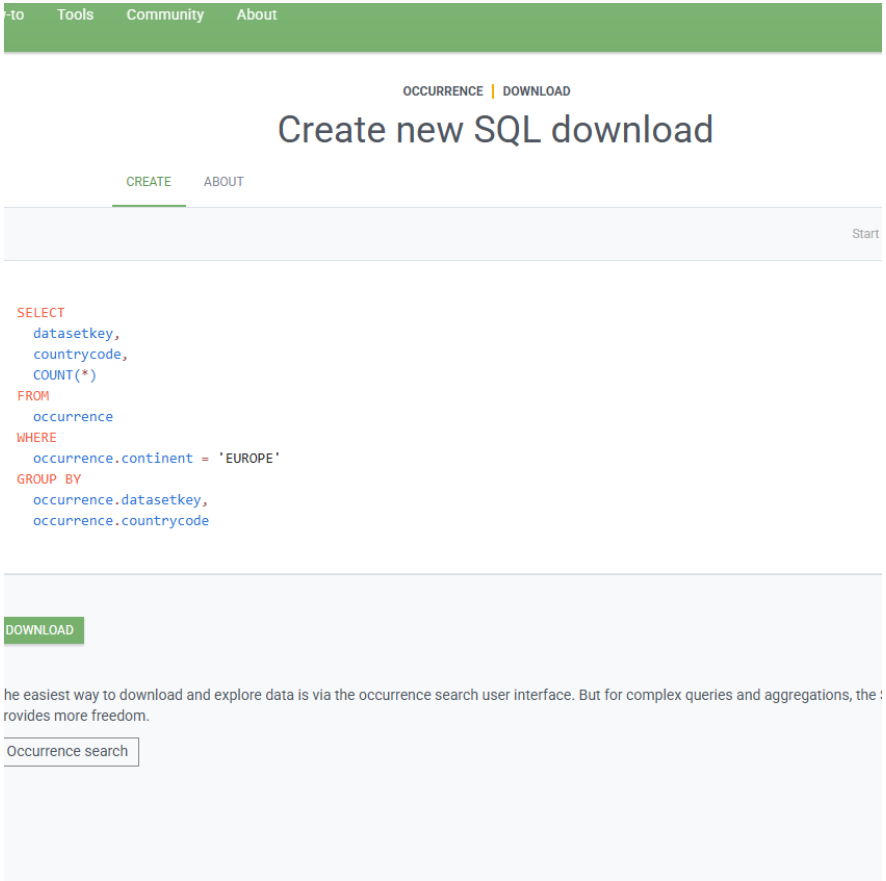
- Simple
- Dwc Archive
- Species List



Funded by the European Union

SQL Downloads

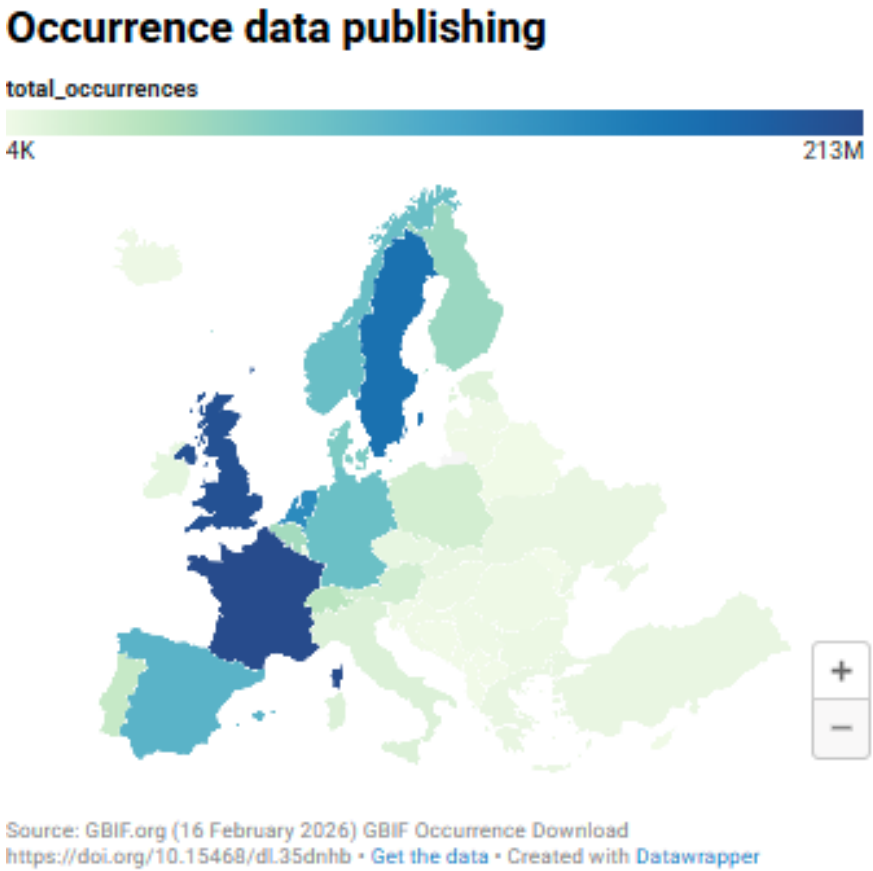
- Occurrence API in GBIF can be queried using SQL - Structured Query Language
- SQL API Interface - https://www.gbif.org/occurrence/download/sql
- Via clients - rgbif, pygbif
- SQL API allows for summary views of GBIF data
- Reduced size of downloads
- **Citable DOI** that gives attribution to all publishers without needing a derived dataset
- Need to have a GBIF account



Funded by the European Union

Example: Regional Summary Counts

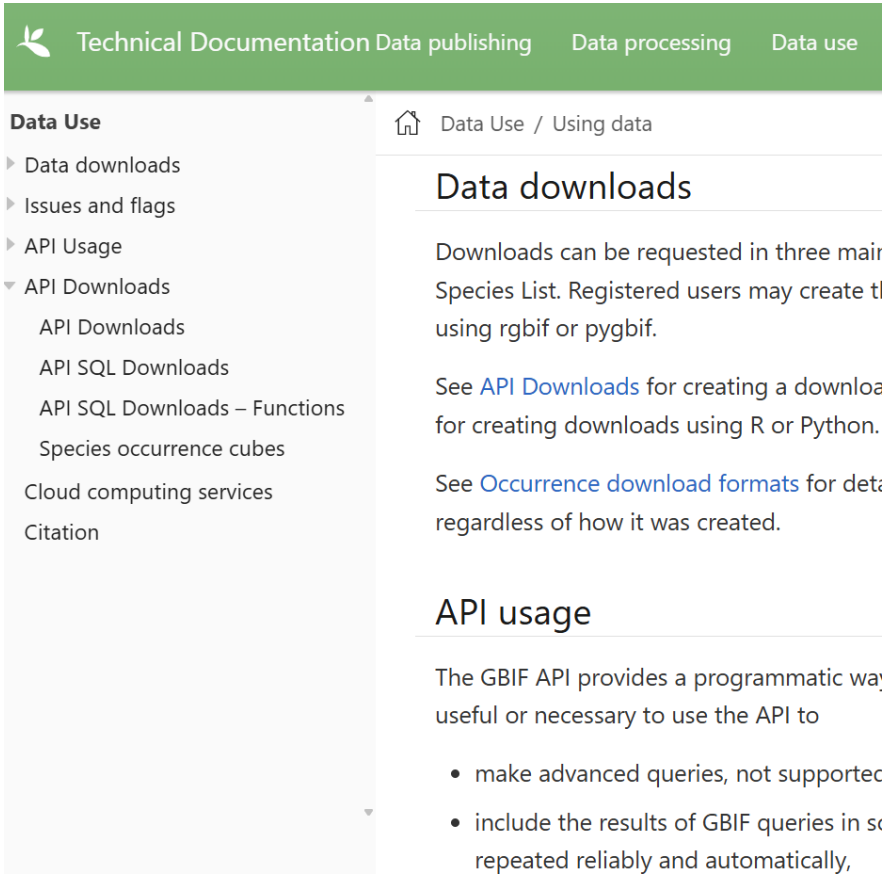
- What is the number of data publishers publishing data about each country in Europe and what is the breakdown in number of species occurrences by kingdom all countries in Europe?
- Group by country
- Summary counts for:
  - Total Occurrences
  - Numbers of publishers
  - Occurrences for each kingdom
- >1.2 billion occurrence records for a normal download
- SQL download - 51 rows and 12 columns



Funded by the European Union

Documentation and training

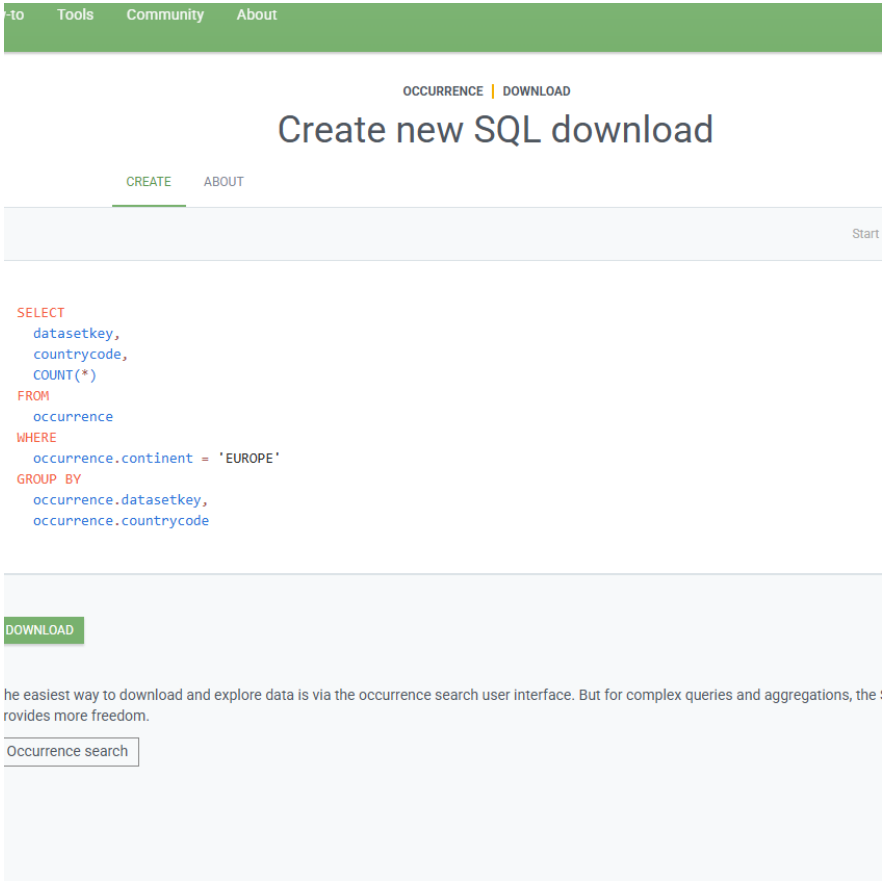
- Technical documentation https://techdocs.gbif.org/en/data-use/api-sql-downloads
- Data Management with SQL for Ecologists – Aggregating and grouping data - https://datacarpentry.github.io/sql-ecology-lesson/02-sql-aggregation.html
- Data Camp “Introduction to SQL” and “Intermediate SQL” cover most of the basics - https://www.gbif.org/data-use-club



Funded by the European Union

What is a SQL query?

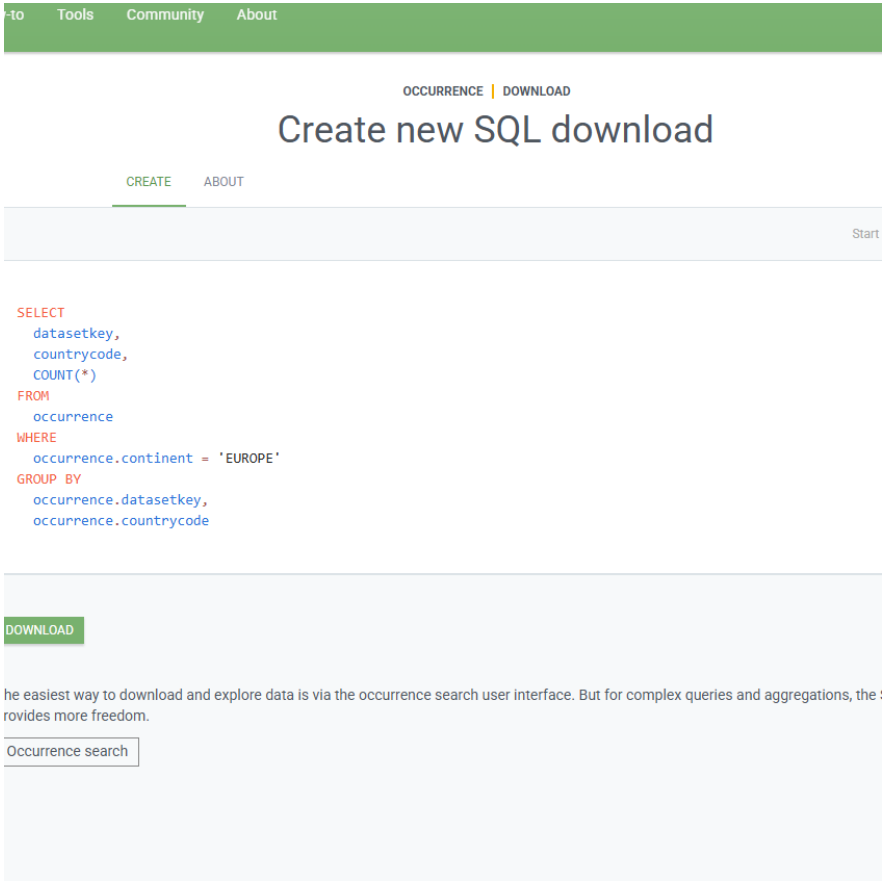
- Queries typically start with SELECT and FROM keywords
- SELECT indicates which fields from the database selected
- FROM indicates the table in which these fields are located
- Operators are capitalized



Funded by the European Union

Filtering

- WHERE keywords allows you to apply filters on your query
- Use WHERE with comparison operators eg =, >, <, >=, =<
- Combine additional features with AND and OR



Funded by the European Union

Column name	Data type	Nullable	Definition
gbifid	String	No	Unique GBIF key for the occurrence. We aim to keep these keys stable, but this is not possible in every case.
accessrights	String	Yes	Information about who can access the resource or an indication of its security status.
bibliographiccitation	String	Yes	A bibliographic reference for

Available columns

- The occurrence table contains 421 columns available for querying:
  - the interpreted data,
  - the verbatim data identified with v_ prefix
  - some calculated columns useful for searching.
- Note the- column names that are also SQL keywords must be quoted. This affects the columns "year", "month", "day", "order", "group", "language", "references " and "member". They must be quoted even if qualified with the table name as in occurrence."year" or "occurrence"."year".

Available columns: https://techdocs.gbif.org/en/data-use/api-sql-downloads#sql-columns

Funded by the European Union

Let’s practice – selecting only columns of interest

- Sometimes we only want a select number of columns and rows e.g. I want all occurrence records for ring-necked parakeets *Psittacula krameri* in Belgium but with only lat-long coordinates and dates
- 117,252 occurrences (rows) in a standard download

```
SELECT
occurrenceid,
eventdate,
decimallatitude,
decimallongitude
FROM
occurrence
WHERE
countrycode = ‘BE'
AND specieskey = ' 2479226 '
```


Funded by the European Union

Let’s practice – selecting only columns of interest

occurrenceid	eventdate	decimallatitude	decimallongitude
INBO:MEETNET:OCC:0128424	25/05/2017	50.72479	4.52185
INBO:MEETNET:OCC:0131991	10/06/2017	50.89708	4.4141
INBO:MEETNET:OCC:0158163	20/06/2017	50.76153	4.46641
INBO:MEETNET:OCC:0489887	05/05/2018	51.25533	4.52475
INBO:MEETNET:OCC:0625157	29/06/2018	50.89799	4.27195
INBO:MEETNET:OCC:0748215	09/07/2020	51.28307	4.46837
INBO:MEETNET:OCC:0749942	15/07/2020	50.72479	4.52185
INBO:MEETNET:OCC:0770543	21/06/2020	50.76153	4.46641
INBO:MEETNET:OCC:0017447	12/03/2017	51.28307	4.46837
INBO:MEETNET:OCC:0790305	01/05/2021	50.93824	4.68524
INBO:MEETNET:OCC:0811920	10/07/2021	50.82368	4.52529
INBO:MEETNET:OCC:1092410	26/05/2024	51.28359	4.42537
INBO:MEETNET:OCC:1092419	26/05/2024	51.28359	4.42537
INBO:MEETNET:OCC:1043755	08/04/2024	50.8217	5.17841
INBO:MEETNET:OCC:1019923	17/06/2023	50.9329	4.429
INBO:MEETNET:OCC:0658950	14/04/2019	50.83979	4.65369
INBO:MEETNET:OCC:0491895	12/05/2018	50.80616	3.1198
INBO:MEETNET:OCC:0723107	19/03/2020	50.76153	4.46641
INBO:MEETNET:OCC:0747393	16/06/2020	50.82407	4.49691
INBO:MEETNET:OCC:0749947	15/07/2020	50.72479	4.52185
INBO:MEETNET:OCC:0805484	14/06/2021	51.07634	4.46136
INBO:MEETNET:OCC:0808319	27/06/2021	50.91492	4.42869
INBO:MEETNET:OCC:1086241	17/05/2024	50.90607	4.41432
INBO:MEETNET:OCC:1104196	07/05/2023	50.82971	4.72432
INBO:MEETNET:OCC:1097162	30/05/2024	50.8217	5.17841
INBO:MEETNET:OCC:0120479	14/04/2017	50.72479	4.52185
INBO:MEETNET:OCC:0493566	25/05/2018	50.82368	4.52529
INBO:MEETNET:OCC:0137729	08/07/2017	50.73358	4.53633

https://doi.org/10.15468/dl.3jeju5

Funded by the European Union

Can include quality assurance filters

```
WHERE
countrycode = ‘PL’
AND occurrencestatus = 'PRESENT’
AND specieskey= 2441184
AND hasgeospatialissues = FALSE
AND NOT GBIF_STRINGARRAYCONTAINS(issue, 'TAXON_MATCH_FUZZY', TRUE)
AND (distancefromcentroidinmeters >= 2000.0 OR distancefromcentroidinmeters IS NULL)
AND NOT basisofrecord IN ('FOSSIL_SPECIMEN', 'LIVING_SPECIMEN')
```

- Only records from **Poland** (PL)
- Only include records where the European bison *Bison bonasus* was identified as being present (specieskey= 2441184)
- Exclude records flagged with coordinate problems
- Exclude records where GBIF had to “guess” the taxonomic match
- Only keep records: that are ≥ 2 km away from centroid OR where centroid distance isn’t recorded
- Remove fossils and living specimens in zoo / botanical garden specimens

Bison bonasus (Linnaeus, 1758) Observed in Poland by Fabian (licensed under http://creativecommons.org/licenses/by-nc/4.0/)

Funded by the European Union

Some common aggregate functions

- COUNT (*) – Count all records (rows), including those with NULL values

```
SELECT
COUNT(*)
FROM
occurrence
WHERE
specieskey = 1423395
AND countrycode = 'SE'
AND "month" IN (10, 11)
```

Funded by the European Union

Some common aggregate functions

- For the blue-tailed damselfly *Ischnura elegans* in Sweden, what were the number of occurrences combined for both October and November.

COUNT(*)
38

https://doi.org/10.15468/dl.yjmnte

```
SELECT
COUNT(*)
FROM
occurrence
WHERE
specieskey = 1423395
AND countrycode = 'SE'
AND "month" IN (10, 11)
```

Funded by the European Union

Some common aggregate functions

```
SELECT
COUNT(individualcount)
FROM
occurrence
WHERE
specieskey = 1423395
AND countrycode = 'SE'
AND "month" IN (10, 11)
```

- COUNT (field_name) – counts non-NULL values for a field

Funded by the European Union

Some common aggregate functions

- For the blue-tailed damselfly *Ischnura elegans* in Sweden in the months of October and November, how many occurrences have non-null values for individualcount?

```
SELECT
COUNT(individualcount)
FROM
occurrence
WHERE
specieskey = 1423395
AND countrycode = 'SE'
AND "month" IN (10, 11)
```

COUNT(individualcount)
25

https://doi.org/10.15468/dl.t88f62

Funded by the European Union

Some common aggregate functions

- COUNT (DISTINCT field_name) – counts number of unique values within a column

```
SELECT
COUNT(DISTINCT specieskey)
FROM
occurrence
WHERE
countrycode = 'SE'
AND "year" = 2025
AND "month" IN (10, 11)
```

Funded by the European Union

Some common aggregate functions

- How many unique species were recorded in Sweden in October and November 2025?

COUNT(DISTINCT specieskey)
9011

```
SELECT
COUNT(DISTINCT specieskey)
FROM
occurrence
WHERE
countrycode = 'SE'
AND "year" = 2025
AND "month" IN (10, 11)
```

https://doi.org/10.15468/dl.kavvpe

Funded by the European Union

Some common aggregate functions

- SELECT DISTINCT (field_name) – retrieves unique values within a column

```
SELECT DISTINCT
specieskey
FROM
occurrence
WHERE
countrycode = 'SE'
AND "year" = 2025
AND "month" IN (10, 11)
```

Funded by the European Union

Some common aggregate functions

- What were the unique species recorded in Sweden in October and November 2025?

- https://doi.org/10.15468/dl.cz425t

specieskey
2493091
2484916
5258334
6092830
6100954
2542720
3368245
2543102
5244293
2520148
9813242
4524054
2484596
2538630
3362604
12123140

```
SELECT DISTINCT
specieskey
FROM
occurrence
WHERE
countrycode = 'SE'
AND "year" = 2025
AND "month" IN (10, 11)
```

Funded by the European Union

GROUPING

- Use GROUP BY clause to group results with aggregate functions to provide summary statistics for each grouping
- It will give one value for the aggregate functions for the combination of fields

```
SELECT
specieskey,
COUNT(*)
FROM
occurrence
WHERE
countrycode = 'SE'
AND "year" = 2025
AND "month" IN (10, 11)
GROUP BY
specieskey
```

Funded by the European Union

GROUPING

- What are the number of occurrences for each species found in Sweden in October and November 2025?

```
SELECT
specieskey,
COUNT(*)
FROM
occurrence
WHERE
countrycode = 'SE'
AND "year" = 2025
AND "month" IN (10, 11)
GROUP BY
specieskey
```

specieskey	COUNT(*)
12265843	5
12264139	1
12262923	1
12261787	93
12261739	4
12261420	1
12261358	1
12260886	2
12256167	5
12247229	60
12245166	5
12243943	7
12243629	1
12242752	1
12242676	34
12239754	40

https://doi.org/10.15468/dl.wsh7er

Funded by the European Union

GROUPING

- What are the number of occurrences for all species recorded for both October and November 2025 in Sweden?

```
SELECT
specieskey,
"month",
COUNT(*)
FROM
occurrence
WHERE
countrycode = 'SE'
AND "year" = 2025
AND "month" IN (10, 11)
GROUP BY
specieskey,
"month"
```

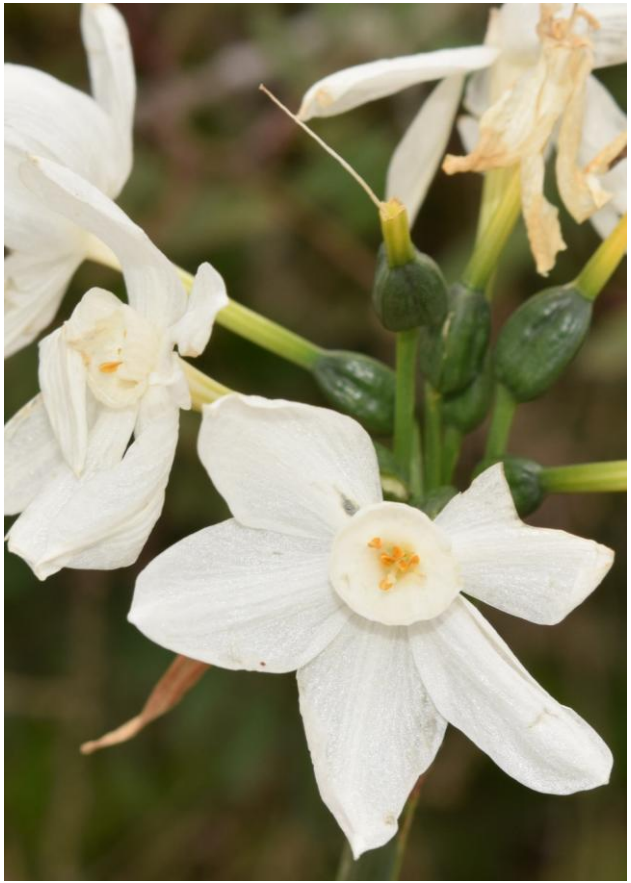
specieskey	month	COUNT(*)
12265843	10	3
12265843	11	2
12264139	10	1
12262923	11	1
12261787	10	93
12261739	10	2
12261739	11	2
12261420	10	1
12261358	11	1
12260886	10	2
12256167	11	2
12256167	10	3
12247229	10	49
12247229	11	11
12245166	11	2
12245166	10	3

https://doi.org/10.15468/dl.jphae6

Funded by the European Union

Let’s practice : identifying trends

- What species of *Narcissus* is most commonly observed in January, February and March in Spain and what have been the relative changes between species between the years 2020 and 2025?

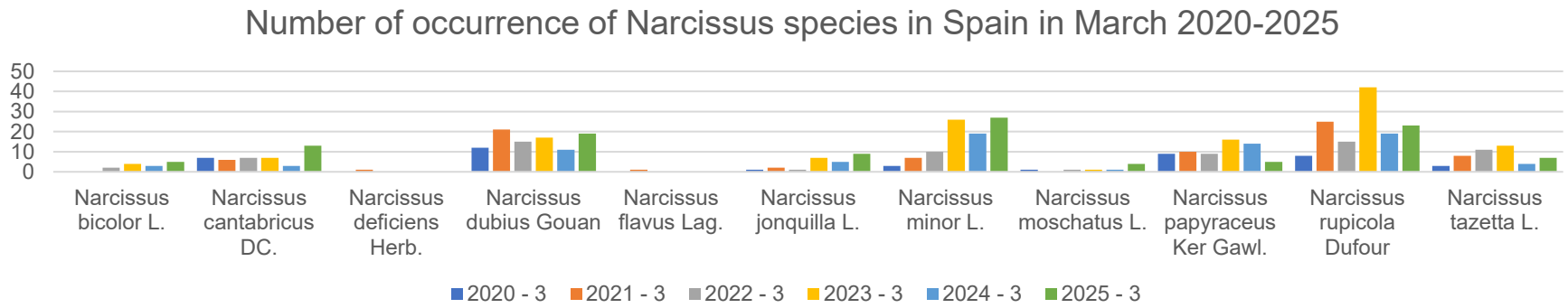


```
SELECT
scientificname,
acceptedscientificname
specieskey,
genuskey,
“year”,
“month”,
COUNT(*)
FROM
occurrence
WHERE
countrycode = ‘SP'
AND genuskey = ' 144107352 ‘
AND “year” >= 2020
AND "year" <= 2025
AND “month” IN (1, 2, 3)
GROUPBY
scientificname
acceptedscientificname,
specieskey,
genuskey,
“year”
“month”
```


Funded by the European Union

scientificname	acceptedscientificname	specieskey	genuskey	year	month	COUNT(*)
Narcissus papyraceus Ker Gaw	Narcissus papyraceus Ker Gawl.	2858739	2858200	2020	1	39
Narcissus tazetta L.	Narcissus tazetta L.	2858963	2858200	2020	1	9
Narcissus dubius Gouan	Narcissus dubius Gouan	2859205	2858200	2020	1	4
Narcissus assoanus Dufour ex S	Narcissus assoanus Dufour ex Schult	2859233	2858200	2020	1	4
Narcissus × splendidus P.Góme	Narcissus × splendidus P.Gómez-Mu	12182342	2858200	2020	1	1
Narcissus cantabricus DC.	Narcissus cantabricus DC.	2858679	2858200	2020	1	5
Narcissus pallidiflorus Pugsley	Narcissus pallidiflorus Pugsley	2858403	2858200	2020	1	3
Narcissus bulbocodium L.	Narcissus bulbocodium L.	2858846	2858200	2020	1	5
Narcissus assoanus subsp. ass	Narcissus assoanus subsp. assoanus	2859233	2858200	2020	1	2
Narcissus tazetta subsp. tazett	Narcissus tazetta subsp. tazetta	2858963	2858200	2020	1	4
Narcissus pseudonarcissus L.	Narcissus pseudonarcissus L.	2858244	2858200	2020	1	1
Narcissus serotinus L.	Narcissus serotinus L.	2858672	2858200	2020	1	2
Narcissus confusus Pugsley	Narcissus confusus Pugsley	2858371	2858200	2020	1	1
Narcissus triandrus subsp. pall	Narcissus triandrus subsp. pallidulus	2858501	2858200	2020	1	1
Narcissus bulbocodium subsp.	Narcissus bulbocodium subsp. bulbo	2858846	2858200	2020	1	1
Narcissus triandrus L.	Narcissus triandrus L.	2858501	2858200	2020	1	1
Narcissus tazetta L.	Narcissus tazetta L.	2858963	2858200	2021	1	19
Narcissus papyraceus Ker Gaw	Narcissus papyraceus Ker Gawl.	2858739	2858200	2021	1	44
Narcissus triandrus L.	Narcissus triandrus L.	2858501	2858200	2021	1	2
Narcissus cantabricus DC.	Narcissus cantabricus DC.	2858679	2858200	2021	1	15
Narcissus tazetta subsp. tazett	Narcissus tazetta subsp. tazetta	2858963	2858200	2021	1	2

https://doi.org/10.15468/dl.zg3un8



Narcissus papyraceus Ker Gawl. Observed in Spain by faluke (licensed under http://creativecommons.org/licenses/by-nc/4.0/)

Funded by the European Union

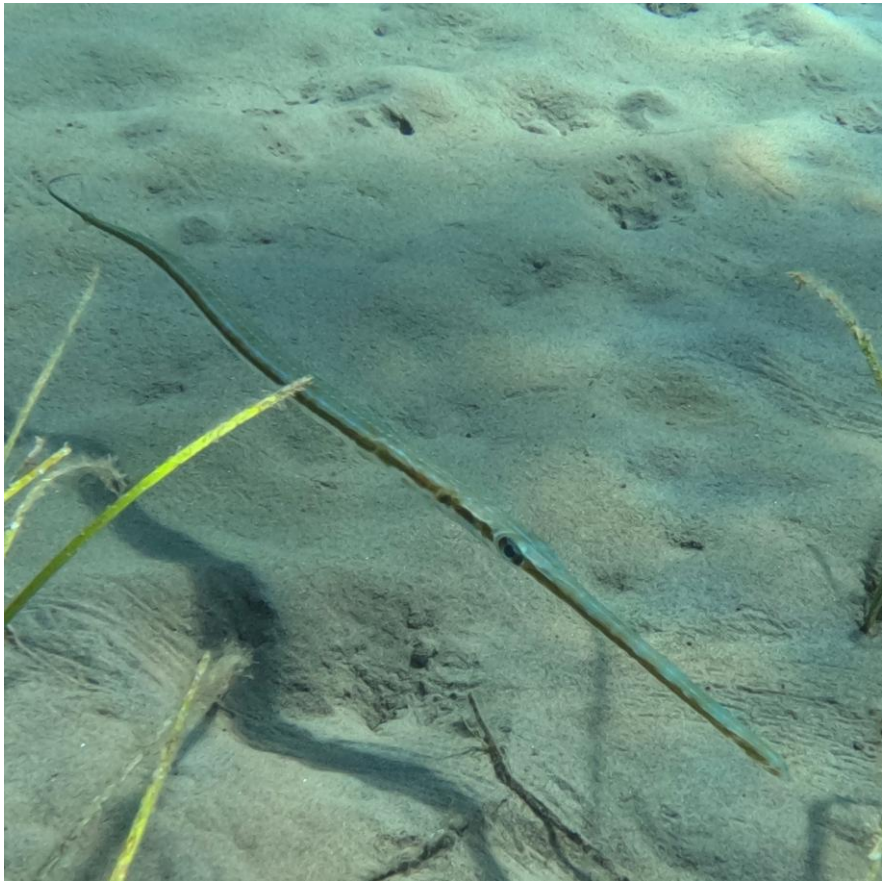
Let’s practice: some new aggregate functions

- For introduced species in Greece, we want to know when it was first recorded, what were the highest number of individuals recorded and how many different years it was recorded after it was first recorded?
- What is the MINIMUM year they were observed in
- What is the MAXIMUM number of individuals observed in a single occurrence
- COUNT the number of DISTINCT years the species was observed in

```
SELECT
scientificname,
specieskey,
establishmentmeans,
MIN("year") earliest_year,
MAX(organismquantity) max_count,
COUNT(DISTINCT "year") years_with_records
FROM
occurrence
WHERE
countrycode = 'GR’
AND occurrencestatus = 'PRESENT’
AND
GBIF_STRINGARRAYCONTAINS(establishmentmeans.line
age, 'introduced', FALSE)
AND organismquantity > 1
AND organismquantitytype = 'individuals’
GROUP BY
scientificname,
specieskey,
establishmentmeans
```

scientificname	specieskey	establishmentmeans	earliest_year	max_count	years_with_records
Callinectes sapidus Rathbun, 1896	2225646	introduced	1959	516	9
Hydroides elegans (Haswell, 1883)	2328302	introduced	1976	1574	3
Ficopomatus enigmaticus (Fauvel, 1923)	2327735	introduced	1976	511	2
Conomurex persicus (Swainson, 1821)	6489949	introduced	1983	2000	11
Pempheris rhomboidea Kossmann & Räuber, 1877	2374391	introduced	1985	100	5
Pinctada radiata (Leach, 1814)	4374764	introduced	1990	500	12
Styela plicata (Lesueur, 1823)	2331954	introduced	1994	660	4
Upeneus moluccensis (Bleeker, 1855)	2396768	introduced	1995	4	2
Fulvia fragilis (Forsskål, 1775)	5729175	introduced	1997	2	3
Siganus luridus (Rüppell, 1829)	2390262	introduced	1998	1223	16
Siganus rivulatus Forsskål & Niebuhr, 1775	2390185	introduced	1998	2000	14
Brachidontes pharaonis (P.Fischer, 1870)	4374596	introduced	1999	100	3
Bursatella leachii Blainville, 1817	9680062	introduced	2001	115	9
Fistularia commersonii Rüppell, 1838	5200938	introduced	2002	245	13
Apogonichthyoides pharaonis (Bellotti, 1874)	5853511	introduced	2002	4	2
Upeneus pori Ben-Tuvia & Golani, 1989	2396797	introduced	2003	12	2
Portunus segnis (Forskål, 1775)	5863622	introduced	2004	4	4
Crepidula fornicata (Linnaeus, 1758)	5192789	introduced	2004	10	3
Etrumeus golanii DiBattista, Randall & Bowen, 2012	8365316	introduced	2005	360	3
Chama pacifica Broderip, 1835	4372070	introduced	2005	7	3
Petricolaria pholadiformis (Lamarck, 1818)	2287615	introduced	2007	10	1
Lagocephalus sceleratus (Gmelin, 1789)	2407758	introduced	2008	30	10
Pterois miles (Bennett, 1828)	2334433	introduced	2008	50	7
Paracerceis sculpta (Holmes, 1904)	2206602	introduced	2009	453	1
Herdmania momus (Savigny, 1816)	5200658	introduced	2010	124	5
Synaptula reciprocans (Forsskål, 1775)	4341995	introduced	2010	30	5
Dendostrea folium (Linnaeus, 1758)	4595289	introduced	2010	10	3
Torquigener flavimaculosus Hardy & Randall, 1983	2407271	introduced	2012	150	7
Sargocentron rubrum (Forsskål, 1775)	2356802	introduced	2012	20	7
Penaeus aztecus Ives, 1891	2223840	introduced	2012	45	6
Diadema setosum (Leske, 1778)	5721108	introduced	2014	480	6
Parupeneus forsskali (Fourmanoir & Guézé, 1976)	2396749	introduced	2017	100	4
Mnemiopsis leidyi A.Agassiz, 1865	2501248	introduced	2018	500	3
Rhopilema nomadica Galil, Spanier & Ferguson, 1990	2264597	introduced	2019	12	1
Stypopodium schimperi (Kützing) Verlaque & Boudouresque	3200580	introduced	2019	1000	1
Spondylus spinosus Schreibers, 1793	5729648	introduced	2020	5	1
Rapana venosa (Valenciennes, 1846)	4363583	introduced		2	0

https://doi.org/10.15468/dl.sfcsgk



Fistularia commersonii Rüppell, 1838 Observed in Greece by tomguckt (licensed under http://creativecommons.org/licenses/by-nc/4.0/)

Funded by the European Union

Custom GBIF SQL Filter Functions - Geospatial distance, GBIF_GeoDistance

Checks if coordinates are within a specified distance of a geographical coordinate- This function returns TRUE if the *point* is within *distance* of the *centroid*.

Parameters:
centroidLatitude -The latitude of the *centroid*, in WGS84 decimal degrees
centroidLongitude - The longitude of the *centroid*, in WGS84 decimal degrees
Distance - A *distance* in metres or kilometres including the unit, e.g. 500m or 2.5km
Latitude - The latitude of the *point* to filter (usually the decimalLatitude field)
Longitude - The longitude of the *point* to filter (usually the decimalLongitude field)

```
SELECT
kingdom,
scientificName,
COUNT(*)
FROM
occurrence
WHERE
GBIF_GeoDistance(56.0, 12.0, '10km',
decimalLatitude, decimalLongitude)
= TRUE
GROUP BY
kingdom,
scientificName;
```


Funded by the European Union

Custom GBIF SQL Filter Functions - Within polygon, GBIF_Within

- Geospatial predicate that checks if the coordinates are inside a polygon. returns TRUE if the *point* is within the *polygon*

- **Parameters:**
- ***geometryWellKnownText -*** A *polygon* specified in Well-known text (WKT) format
- ***Latitude -*** The latitude of the *point* to filter (usually the decimalLatitude field)
- ***Longitude -*** The longitude of the *point* to filter (usually the decimalLongitude field)

- **Notes:**
- If you have problems, first check your polygon with WKT Map or Wicket — ensure the points are ordered anti-clockwise, and are in longitude-latitude order, and form a closed loop with the first and last points equal. (A polygon with clockwise points represents the opposite: the whole world except the polygon.)

```
SELECT
kingdom,
scientificName,
COUNT(*)
FROM
occurrence
WHERE
GBIF_Within('POLYGON ((-64.8 32.3, -65.5 18.3, -80.3 25.2, -64.8 32.3))', decimalLatitude, decimalLongitude) = TRUE
GROUP BY
kingdom,
scientificName;
```


Funded by the European Union

Custom GBIF SQL Filter Functions - GBIF_StringArrayContains

- For working with string arrays. This function returns TRUE if one or more elements in the array match the query string.

- **Parameters:**
- ***Array -*** A string array
- ***query -*** A string to test against each element of the array
- ***matchCase-*** Whether to match letter case
- **Return value:**
- **Notes:**
- This is similar to the standard ARRAY_CONTAINS function, except for the addition of the matchCase parameter.

```
SELECT
kingdom,
scientificName,
COUNT(*)
FROM
occurrence
WHERE
GBIF_StringArrayContains(occurrenc
e.recordedby, 'Matthew', FALSE)
GROUP BY
kingdom,
scientificName;
```


Funded by the European Union

Custom GBIF SQL Filter Functions - GBIF_StringArrayLike

- LIKE operator for string arrays. Returns TRUE if one or more elements in the array match the query pattern.

- **Parameters:**
- ***array -*** A string array
- ***query-*** A pattern to test against each element of the array, with ? matching a single character and * matching zero or more characters.
- ***matchCase -*** Whether to match letter case

```
SELECT
kingdom,
scientificName,
COUNT(*)
FROM
occurrence
WHERE
GBIF_StringArrayLike(occurrence.recordedby, 'Mat*hew', FALSE)
GROUP BY
kingdom,
scientificName;
```


Funded by the European Union

Features of GBIF SQL QUERIES

- Most common SQL operators, such as AND, OR, NOT, IS NULL, RAND(), ROUND(...), LOWER(...), etc.
- Only SELECT queries. Selecting * not allowed, columns should be specified
- GROUP BY queries allowed as well as basic SQL window functions (OVER and PARTITION BY)
- Aggregate functions, for example COUNT(...), MIN(...), MAX(...), AVERAGE(...)
- Custom filter function eg GBIF_GeoDistance and handling data types GBIF_TemporalUncertainty and custom gridding functions GBIF_EEARGCode
- JOIN queries and sub –queries not allowed
- The group and partition filters (HAVING and QUALIFY) are not yet supported.
- BETWEEN is not supported, use >, >=, < and ⇐ instead.
- Can use ORDER BY

Funded by the European Union

Troubleshooting

- If column names are quoted, they must use double quotes and lowercase: "year", "basisofrecord". It's only necessary to quote column names that are also SQL functions: "year", "month", "day", "order", "group", "language", "references" and "member".
- When column and function names aren't quoted, the letter case doesn't matter: basisOfRecord, basisofrecord, gbif_eeargcode(...), GBIF_EEARGCode(...).
- String values use single quotes, like basisOfRecord = 'HUMAN_OBSERVATION'
- For examples of how to write values within your queries refer to GBIF Occurrence API documentation - https://techdocs.gbif.org/en/openapi/v1/occurrence#/Searching%20occurrences
- SQL comments (lines beginning --) cannot be included in queries.
- Columns with numeric types must be filtered using numbers, not quoted string values — use taxonkey IN (1234, 5687) or "year" >= 2020.
- Array values (including vocabulary lineages) should be converted to delimited strings, for example CONCAT_WS(' | ', recordedByID) and CONCAT_WS(' | ', occurrence.lifestage.lineage).

Funded by the European Union

Passerines in Europe

- Which species of the bird order Passerifomes have the most occurrences in Europe per month? Human observations only.

- Create your own SQL query using a SELECT, FROM, WHERE and GROUP BY operators
- You will need the following filters:
  - orderkey (729)
  - basisofrecord
  - continent
  - specieskey

  (Tip : you will need to use an IS NOT NULL operator with one of your filters)

Sitta europaea Linnaeus, 1758 .Observed in Sweden by Joel Q (licensed under http://creativecommons.org/licenses/by-nc/4.0/)

Funded by the European Union

Passerines in Europe

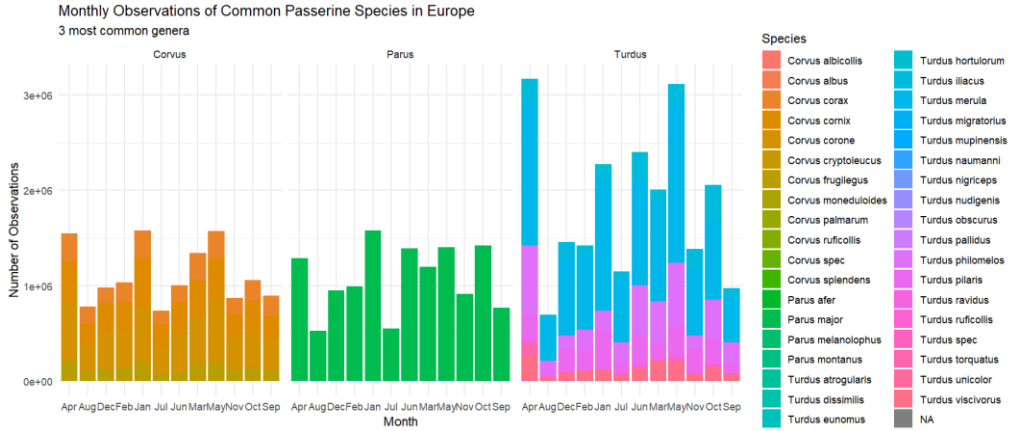
```
SELECT
specieskey,
"month",
COUNT(*)
FROM
 occurrence
WHERE
basisofrecord = 'HUMAN_OBSERVATION'
AND occurrence.orderkey = '729'
AND occurrence.continent = 'EUROPE'
AND occurrence.specieskey IS NOT NULL
GROUP BY
specieskey,
"month"
```

Funded by the European Union

Turdus merula Linnaeus, 1758 Observed in United Kingdom of Great Britain and Northern Ireland by chrismalumphy (licensed under http://creativecommons.org/licenses/by-nc/4.0/)

Parus major Linnaeus, 1758 Observed in Spain by mickeydylan (licensed under http://creativecommons.org/licenses/by-nc/4.0/)

https://doi.org/10.15468/dl.w3rkkd

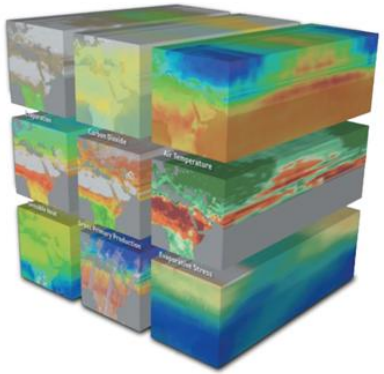


https://gist.github.com/PietrH/0f544d4242667383160b98772dafe47b

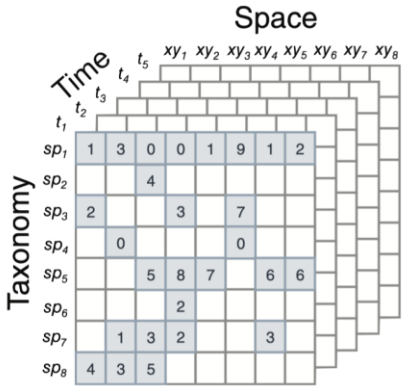
Funded by the European Union


FROM SQL TO SPECIES
OCCURRENCE CUBES.....

Data Cubes



Visualisation of an Earth system data cube by ESA.



- A **data cube** is a multidimensional dataset that allows for fast slicing and dicing along key dimensions and is commonly used in in the climate modelling and remote sensing communities

- GBIF-mediated data can be structured into a cube format along three dimensions
  - Taxonomy
  - Spatial
  - Temporal
- Creates a species occurrence cube

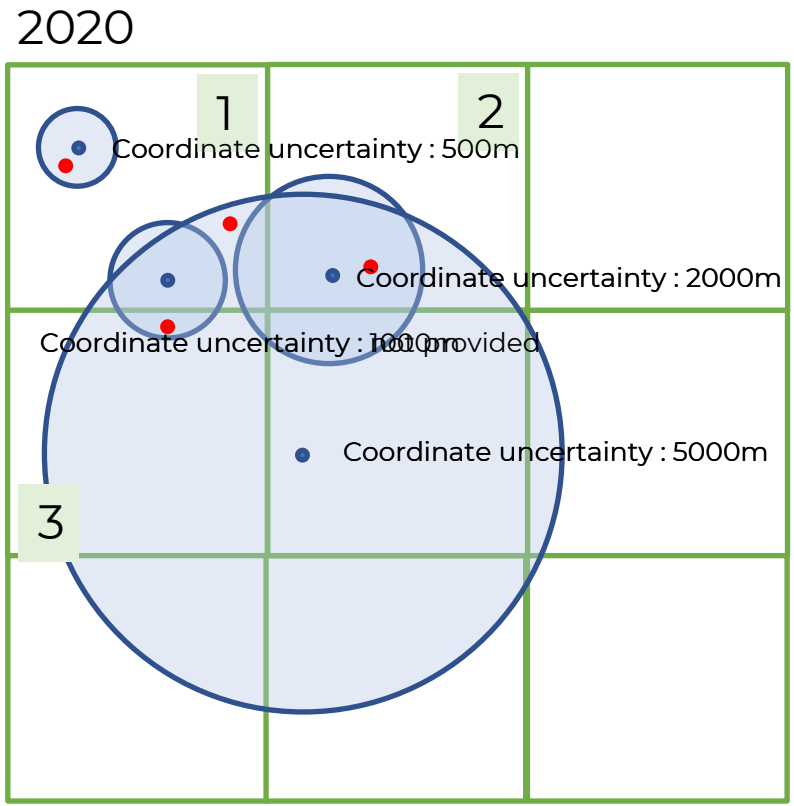
Funded by the European Union

Species Occurrence Cubes

- Uses SQL download capabilities to create summaries of GBIF mediated associated to user-defined spatial reference grids
  - European Environment Agency Reference Grid (EEA)
  - Extended Quarter Degree Grid (EQDG)
  - Inverse Snyder Equal-Area Projection Aperture 3 Hexagonal (ISEA3H)
  - Discrete Global Grid System (DGGS),
  - Military Grid Reference System (MGRS)
  - Degree-Minute -Second Grid (DMS)
- Assigns occurrence to grid
  - Take into account uncertainty
  - Random assignment within uncertainty

Funded by the European Union

Aggregation Of Occurrences To Grid Cells

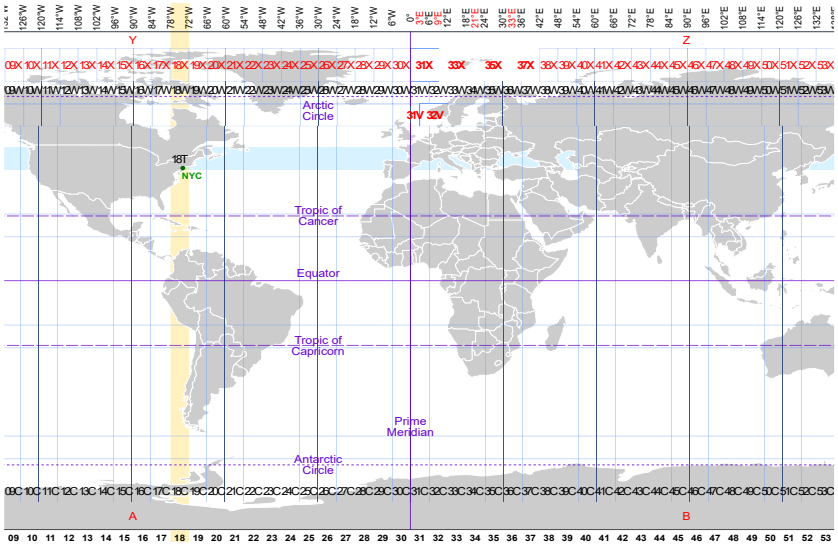


- Occurrence records come with coordinate uncertainties
- Where coordinate uncertainty is not provided default of 1000m is attributed to occurrence
- Point is randomly chosen within uncertainty boundary and the occurrence is assigned to the cell
- In this example, grid 1 = 2 occurrences, grid 2 = 1 occurrence and grid 3 occurrence = 1 occurrence

Funded by the European Union

Grid functions

Grid System	Type	Best For	Key Features
EEA Reference Grid	Square	**European biodiversity monitoring**	Standardized across Europe, Lambert equal-area projection
Extended Quarter-Degree Grid (EQDG)*	Square	**Global species distribution**	15-minute (\u00bc-degree) grid cells, used in ecological studies
Degree-Minute-Second (DMS) Grid	Coordinate System	**Fine-scale species mapping**	Not a grid, but a coordinate system for high-precision GPS locations
ISEA3H Grid (Hexagonal)	Hexagonal	**Climate & ecosystem modeling**	Avoids distortion, equal-area representation, used in global studies
Military Grid Reference System (MGRS)	Square	**Field conservation**	Hierarchical, GPS-compatible, used in military & emergency response



*Not supported on user interface

Funded by the European Union

Custom Grid Functions

Functions comprise of similar parameters:

- Resolution
- latitude of the point to grid (usually the decimalLatitude field)
- The longitude of the point to grid (usually the decimalLongitude field)
- coordinateUncertaintyInMeters - uncertainty radius to apply to the point (usually used with a default value, for example COALESCE(coordinateUncertaintyInMeters, 1000)). The COALESCE function ensures there's always a usable value by replacing NULLs with 1000. Set to 0 to disable randomization.

Each of the functions will give you the code the cell of the grid which the point falls

More information on how to map cube outputs can be found at our Mapping Occurrence Maps Data Use Club Session - https://www.gbif.org/event/7W6Cm2zvuCHnjqelDmWcD/data-use-club-practical-session-making-occurrence-maps

Funded by the European Union

Custom Grid Functions - examples - EEARGCode

```
SELECT
        kingdom,
        scientificName,
        COUNT(*),
        GBIF_EEARGCode(10000,decimalLatitude,decimalLongitude,COALESCE(coordinateUncertaintyInMeters,1000) )
FROM
        occurrence
WHERE
        country = 'PL'
GROUP BY
        kingdom,
        scientificName
```

Result	*gridSize*	*latitude*	*longitude*	*coordinateUncertaintyInMeters*
100kmE43N32	100 000	52.0	10.0	1000.0
50kmE425N265	50 000	52.0	10.0	1000.0
10kmE432N321	10 000	52.0	10.0	1000.0
1kmE4321N3210	1 000	52.0	10.0	1000.0
250mE432100N321000	250	52.0	10.0	1000.0
100mE43210N32100	100	52.0	10.0	1000.0
25mE4321000N3210000	25	52.0	10.0	1000.0

GeoPackage and ShapeFile downloads of the grids at several resolutions are available for download at the European Environment Agency Datahub

Funded by the European Union

Custom Grid Functions - examples - GBIF_EQDGCode

```
SELECT
        kingdom,
        scientificName,
        COUNT(*),
        GBIF_EQDGCode(2,decimalLatitud
        e,decimalLongitude,COALESCE(co
        ordinateUncertaintyInMetres,1000
        ) )
FROM
        occurrence
WHERE
        country = 'ZA'
GROUP BY
        kingdom,
        scientificName
```

Result	*level*	*latitude*	*longitude*	*coordinateUncertaintyInMeters*
E010N52	0	52.3	10.3	1000.0
E010N52C	1	52.3	10.3	1000.0
E010N52CB	2	52.3	10.3	1000.0
E010N52CBC	3	52.3	10.3	1000.0
E010N52CBCC	4	52.3	10.3	1000.0
E010N52CBCCB	5	52.3	10.3	1000.0
E010N52CBCCBB	6	52.3	10.3	1000.0

GeoPackage downloads of the EQDGC grid at levels 0–7 are available for download from GBIF. These were generated using this Java code.

Funded by the European Union

Custom Grid Functions - examples - GBIF_DMSGCode

```
SELECT
        kingdom,
        scientificName,
        COUNT(*),
        GBIF_DMSGCode(900,decimalLatitude,decimalLongitude,COALESCE(coordinateUncertaintyInMeters,1000) )
FROM
        occurrence
WHERE
        country = 'ZA'
GROUP BY
        kingdom,
        scientificName;
```

Result	*level*	*latitude*	*longitude*	*coordinateUncertaintyInMeters*
E010°N52° (E010°N52°)	3600 (1°)	52.3	10.8127	1000.0
E010°45'N52°15'(E010°45'N52°15')	900 (15')	52.3	10.8127	1000.0
E010°40'N52°10'(E010°40'N52°10')	600 (10')	52.3	10.8127	1000.0
E010°45'N52°15'(E010°45'N52°15')	300 (5')	52.3	10.8127	1000.0
E010°47'30"N52°17'30"(E010°47'30"N52°17'30")	150 (2½')	52.3	10.8127	1000.0
E010°48'N52°18'(E010°48'N52°18')	60 (1')	52.3	10.8127	1000.0
E010°48'30"N52°18'00"(E010°48'30"N52°18'00")	30	52.3	10.8127	1000.0
E010°48'45"N52°18'00"(E010°48'45"N52°18'00")	1	52.3	10.8127	1000.0

GeoPackage downloads of a Degree-Minute-Second grid at 3600" (1°), 1800 (30'), 900" (15'), 600" (10'), 300" (5'), 150" (2'30"), 60" (1') and 30" (30") are available for download from GBIF. These were generated using this Java code.

Custom Grid Functions - examples - GBIF_ISEA3HCode

```
SELECT
        kingdom,
        scientificName,
        COUNT(*),
        GBIF_ISEA3HCode(6,decimalLatitude,decimalLongitude,COALESCE(coordinateUncertaintyInMeters, 1000) )
FROM
        occurrence
WHERE
        continent = 'AFRICA'
GROUP BY
        kingdom,
        scientificName
```

Result	*resolution*	*latitude*	*longitude*	*coordinateUncertaintyInMeters*
-358282526011250000	3	52.3	10.3	1000.0
652180731009071912	6	52.3	10.3	1000.0
952458899010519815	9	52.3	10.3	1000.0

GeoPackage downloads of the ISEA3H grid at levels 0–11 are available for download from GBIF.

Funded by the European Union

Custom Grid Functions - examples - GBIF_MGRSCode

```
SELECT
        kingdom,
        scientificName,
        COUNT(*),
        GBIF_MGRSCode(10000,decimalLatitude,decimalLongitude,COALESCE(coordinateUncertaintyInMeters, 1000) )
FROM
        occurrence
WHERE
        country = 'CG'
GROUP BY
        kingdom,
        scientificName;
```

Result	*gridSize*	*latitude*	*longitude*	*coordinateUncertaintyInMeters*
32U	0	52.0	10.0	1000.0
32UNC	100 000	52.0	10.0	1000.0
32UNC66	10 000	52.0	10.0	1000.0
32UNC6861	1 000	52.0	10.0	1000.0
32UNC686615	100	52.0	10.0	1000.0
32UNC68646151	10	52.0	10.0	1000.0
32UNC6864961510	1	52.0	10.0	1000.0

Shapefile downloads of MGRS areas are available from Office of Geomatics (USA National Geospatial-Intelligence Agency) and Geodatabase format downloads are available at mgrs-data.org.

Funded by the European Union

Handling temporal aspects of a cube

- The default query to generate the cubes ensures the included occurrences have sufficient date resolution to fit in the required dimension, so:
    - a year cube needs occurrences with a specific year (year != null).
    - a year-month cube needs the month not to be null. An occurrence from 2021-01-01/30 (30 days in January) is included, although 2021-01-31/2021-02-01 (2 days) is excluded.
- Users could adjust this if they want to, e.g. to assign wider ranges to a random day within that range. We haven't provided an SQL function to make this easier.
- Additional custom functions for time are:
    - **GBIF_TemporalUncertainty -** returns the uncertainty in seconds of the date.
    - **GBIF_MillisecondsToISO8601 -** function formats a timestamp to a string like 2024-01-26T13:43:08Z. The UTC timezone (Z) is used.
    - **GBIF_SecondsToISO8601 -** formats a timestamp to a string like 2024-01-26T13:43:08Z. The UTC timezone (Z) is used.
    - **GBIF_SecondsToLocalISO8601 -** formats a timestamp to a string like 2024-01-26T13:43:08. No timezone is included.

Funded by the European Union

Let´s practice

- Perform an occurrence search on gbif.org
- Use the cube download interface
- Examine and adapt underlying code

Year	eea_cell_code	speciesKey	n	min_coord_uncertainty
2000	1km E3809N3113	2889173	1	700
2000	1kmE3809N3135	2889173	1	700
… … … …				
2006	1kmE3936N3071	2889173	1	49
2006	1kmE3947N3132	2889088	1	700
… … … …				
2010	1kmE3883N3121	4038485	1	700
2010	1kmE3884N3121	2889173	1	10
… … … …				
2014	1kmE3886N3121	2889173	51	10
2014	1kmE3886N3122	2889173	109	10
… … … …				
2018	1kmE4047N3067	2889173	1	2828

Funded by the European Union

Thank you!

Andrew Rodrigues
arodrigues@gbif.org

helpdesk@gbif.org

Useful links

SQL Downloads: https://www.gbif.org/occurrence/download/sql
SQL Download documentation: https://techdocs.gbif.org/en/data-use/api-sql-downloads
SQL downloads in rgbif - https://docs.ropensci.org/rgbif/articles/gbif_sql_downloads.html
Occurrence API documentation: https://techdocs.gbif.org/en/openapi/v1/occurrence#/Searching%20occurrences
Making Occurrence Maps: https://www.gbif.org/event/7W6Cm2zvuCHnjqelDmWcD/data-use-club-practical-session-making-occurrence-maps
Data Management with SQL for Ecologists – Aggregating and grouping data - https://datacarpentry.github.io/sql-ecology-lesson/02-sql-aggregation.html
Data Camp "Introduction to SQL" and "Intermediate SQL" - https://www.gbif.org/data-use-club
Understanding taxon keys: https://discourse.gbif.org/t/understanding-gbif-taxonomic-keys-usagekey-taxonkey-specieskey/3045

b-cubed.eu **@BCubedProject** **B-Cubed Project** **@b-cubed.eu**

Biodiversity Building Blocks for policy

This project receives funding from the European Union's Horizon Europe Research and Innovation Programme (ID No 101059592). Views and opinions expressed are those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the EU nor the EC can be held responsible for them.

Funded by the European Union