B SQL Primer for the OpenAlex Snapshot

R. Heller · Date: 2026-05-09 · Last changed: 2026-05-11

B.1 When to use the snapshot

The OpenAlex API (Chapter 5) is ideal for moderate queries. For analyses requiring millions of records — full-database bibliographic coupling, global co-authorship networks, or exhaustive field delineation — the OpenAlex snapshot is more practical. The snapshot is a full database dump released monthly in compressed JSON-lines format.

B.2 Loading the snapshot

OpenAlex provides scripts to load the snapshot into PostgreSQL. After loading:

```
-- Count total works
SELECT COUNT(*) FROM openalex.works;

-- Recent articles with citation counts
SELECT id, display_name, publication_year, cited_by_count
FROM openalex.works
WHERE type = 'article'
  AND publication_year >= 2020
ORDER BY cited_by_count DESC
LIMIT 100;
```

B.3 Common queries

B.3.1 Publication counts by year

```
SELECT publication_year, COUNT(*) AS n_works
FROM openalex.works
WHERE type = 'article'
GROUP BY publication_year
ORDER BY publication_year;
```

B.3.2 Top institutions by output

```
SELECT i.display_name, COUNT(DISTINCT w.id) AS n_works
FROM openalex.works w
JOIN openalex.works_authorships wa ON w.id = wa.work_id
JOIN openalex.works_authorships_institutions wai
  ON wa.work_id = wai.work_id AND wa.author_position = wai.author_position
JOIN openalex.institutions i ON wai.institution_id = i.id
WHERE w.publication_year = 2023
  AND w.type = 'article'
GROUP BY i.display_name
ORDER BY n_works DESC
LIMIT 20;
```

B.3.3 Co-authorship pairs

```sql
SELECT a1.author_id AS author_a,
       a2.author_id AS author_b,
       COUNT(DISTINCT a1.work_id) AS n_collabs
FROM openalex.works_authorships a1
JOIN openalex.works_authorships a2
  ON a1.work_id = a2.work_id AND a1.author_id < a2.author_id
GROUP BY a1.author_id, a2.author_id
HAVING COUNT(DISTINCT a1.work_id) >= 5
ORDER BY n_collabs DESC
LIMIT 100;
```

B.4 Connecting from R

```r
library(DBI)
library(RPostgres)

con <- dbConnect(
  Postgres(),
  dbname = "openalex",
  host = "localhost",
  port = 5432,
  user = "openalex"
)

top_journals <- dbGetQuery(con, "
  SELECT s.display_name, COUNT(*) AS n_works
  FROM openalex.works w
  JOIN openalex.sources s ON w.primary_location_source_id = s.id
  WHERE w.publication_year = 2023
  GROUP BY s.display_name
  ORDER BY n_works DESC
  LIMIT 20
")

dbDisconnect(con)
```

B.5 Performance tips

- **Index key columns**: `work_id`, `author_id`, `institution_id`, `publication_year`, and `cited_by_count` should all be indexed.
- **Partition by year**: For time-series queries, partitioning the works table by `publication_year` speeds up range scans substantially.
- **Materialised views**: Pre-compute expensive joins (e.g., institution–work counts) as materialised views and refresh them after each snapshot update.
- **Use COPY for bulk loads**: The `COPY` command is orders of magnitude faster than row-by-row `INSERT`.