

## **Parcial 2**

Julián Andrés Guisao Fernández

Escuela de ingenierías, Universidad Pontificia Bolivariana

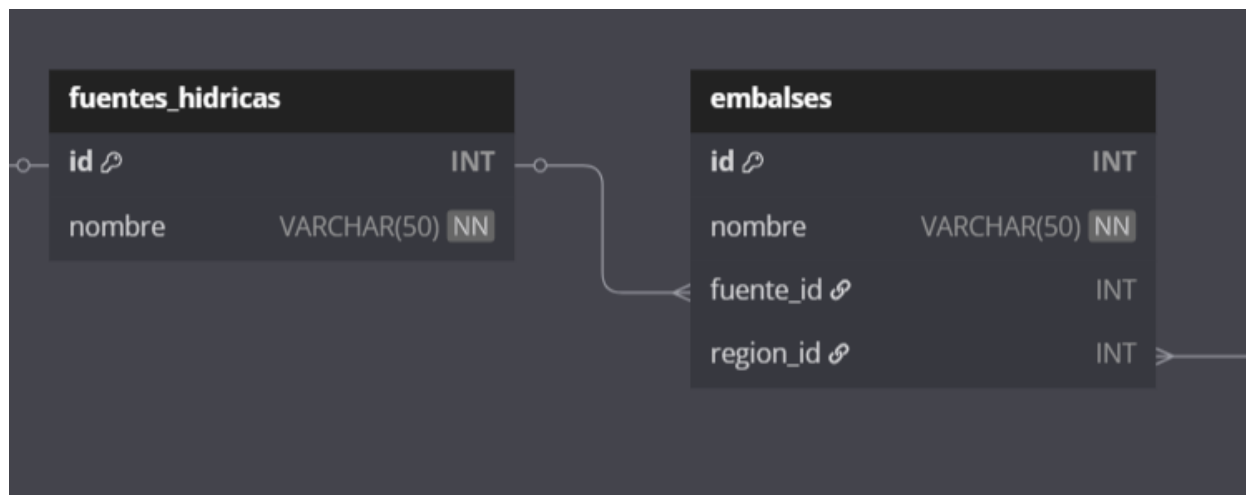
Tópicos Avanzados en Bases de datos

Juan Rodas

3 de abril de 2025

### Etapa 1: Obtención colaborativa de datos faltantes. 20%

Para la normalización de los datos, se identificó que la columna "serie\_hidrologica" representa embalses y fuentes hídricas. Se construyeron dos nuevas tablas: embalses y fuentes\_hidricas.



Creación de tabla de embalses y fuentes hídricas

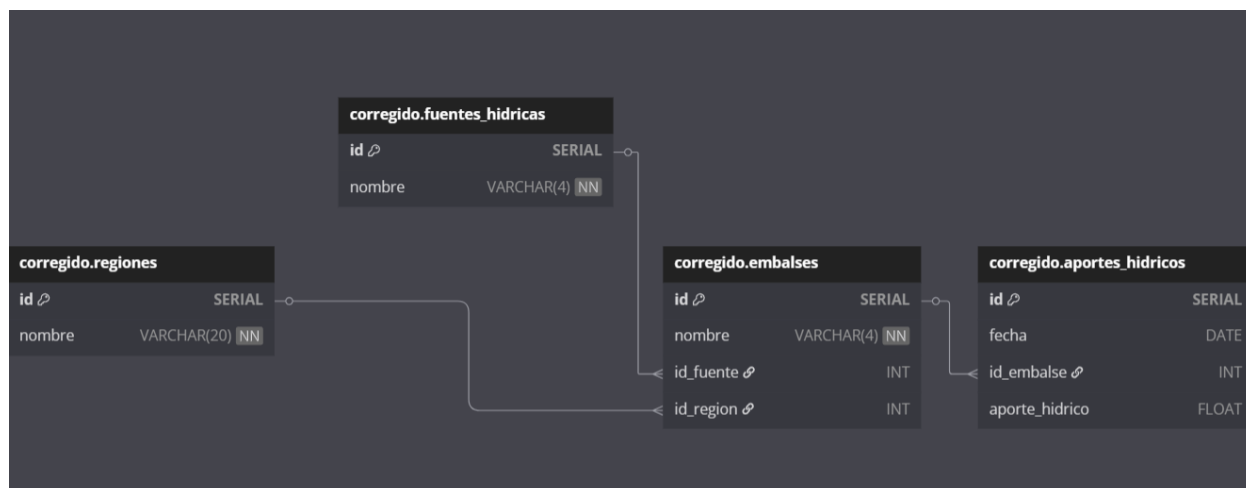
```

CREATE TABLE IF NOT EXISTS corregido.fuentes_hidricas (
  "id" SERIAL PRIMARY KEY,
  "nombre" VARCHAR(4) NOT NULL
);

CREATE TABLE IF NOT EXISTS corregido.embalses (
  "id" SERIAL PRIMARY KEY,
  "nombre" VARCHAR(4) NOT NULL,
  "id_fuente" INT,
  "id_region" INT,
  FOREIGN KEY (id_fuente) REFERENCES corregido.fuentes_hidricas ("id"),
  FOREIGN KEY (id_region) REFERENCES corregido.regiones ("id")
);
  
```

### Etapa 2: Diseño e implementación del modelo de datos – 20%

Para responder consultas de manera eficiente, se diseñó un modelo relacional en tercera forma normal (3FN). A continuación, se muestra la estructura de las tablas y su código de implementación.



Creación de tablas adicionales para completar el esquema base.

```

CREATE SCHEMA IF NOT EXISTS corregido;

CREATE TABLE IF NOT EXISTS corregido.regiones (
  "id" SERIAL PRIMARY KEY,
  "nombre" VARCHAR(20) NOT NULL UNIQUE
);

CREATE TABLE IF NOT EXISTS corregido.fuentes_hidricas (
  "id" SERIAL PRIMARY KEY,
  "nombre" VARCHAR(4) NOT NULL
);

CREATE TABLE IF NOT EXISTS corregido.embalses (
  "id" SERIAL PRIMARY KEY,
  "nombre" VARCHAR(4) NOT NULL,
  "id_fuente" INT,
  "id_region" INT,
  FOREIGN KEY (id_fuente) REFERENCES corregido.fuentes_hidricas ("id"),
  FOREIGN KEY (id_region) REFERENCES corregido.regiones ("id")
);

CREATE TABLE IF NOT EXISTS corregido.aportes_hidricos (
  "id" SERIAL PRIMARY KEY,
  "fecha" DATE,
  "id_embalse" INT,
  "aporte_hidrico" FLOAT,
  FOREIGN KEY (id_embalse) REFERENCES corregido.embalses ("id")
);
  
```

Creación de índices

```

CREATE INDEX IF NOT EXISTS idx_regiones_nombre ON corregido.regiones(nombre);
CREATE INDEX IF NOT EXISTS idx_regiones_id ON corregido.regiones(id);
CREATE INDEX IF NOT EXISTS idx_embalses_nombre ON corregido.embalses(nombre);
CREATE INDEX IF NOT EXISTS idx_embalses_id ON corregido.embalses(id);
CREATE INDEX IF NOT EXISTS idx_fuentes_nombre ON corregido.fuentes_hidricas(nombre);
CREATE INDEX IF NOT EXISTS idx_fuentes_id ON corregido.fuentes_hidricas(id);
CREATE INDEX IF NOT EXISTS idx_aportes_id ON corregido.aportes_hidricos(id);
CREATE INDEX IF NOT EXISTS idx_aportes_fecha ON corregido.aportes_hidricos(fecha);

```

### Etapas 3: Diagnóstico de completitud de datos – 30%

Antes de índices

```

95
96 EXPLAIN ANALYZE
97 WITH dias_por_anio AS (
98     SELECT 2023 AS anio, 365 AS total_dias UNION ALL
99     SELECT 2024 AS anio, 366 AS total_dias
100 ),
101 conteo_aportes AS (
102     SELECT
103         e.id AS id_embalse,
104         e.id_region,
105         EXTRACT(YEAR FROM ah.fecha) AS anio,
106         COUNT(ah.id) AS dias_con_aporte
107     FROM corregido.aportes_hidricos ah
108     JOIN corregido.embalses e ON ah.id_embalse = e.id
109     GROUP BY e.id, e.id_region, anio
110 )
111 SELECT
112     r.nombre AS region,
113     e.nombre AS embalse,
114     c.anio,
115     c.dias_con_aporte,
116     d.total_dias,
117     ROUND((c.dias_con_aporte::NUMERIC / d.total_dias) * 100, 2) AS porcentaje_completitud
118 FROM conteo_aportes c
119 JOIN dias_por_anio d ON c.anio = d.anio
120 JOIN corregido.embalses e ON c.id_embalse = e.id
121 JOIN corregido.regiones r ON e.id_region = r.id
122 ORDER BY r.nombre, e.nombre, c.anio;
123

```

#### QUERY PLAN

1	Sort (cost=3459.39..3460.87 rows=590 width=154) (actual time=29.630..29.638 rows=88 loops=1)
2	Sort Key: r.nombre, e.nombre, (EXTRACT(year FROM ah.fecha))
3	Sort Method: quicksort Memory: 30kB
4	-> Hash Join (cost=1793.63..3432.24 rows=590 width=154) (actual time=29.305..29.537 rows=88 loops=1)
5	Hash Cond: (e.id_region = r.id)
6	-> Hash Join (cost=1763.38..3393.06 rows=590 width=68) (actual time=29.264..29.463 rows=88 loops=1)
7	Hash Cond: (e_1.id = e.id)
8	-> Hash Join (cost=1722.78..3350.91 rows=590 width=48) (actual time=29.173..29.356 rows=88 loops=1)

## Después de índices

```

EXPLAIN ANALYZE
WITH dias_por_anio AS (
  SELECT 2023 AS anio, 365 AS total_dias UNION ALL
  SELECT 2024 AS anio, 366 AS total_dias
),
conteo_aportes AS (
  SELECT
    e.id AS id_embalse,
    e.id_region,
    EXTRACT(YEAR FROM ah.fecha) AS anio,
    COUNT(ah.id) AS dias_con_aporte
  FROM corregido.aportes_hidricos ah
  JOIN corregido.embalses e ON ah.id_embalse = e.id
  GROUP BY e.id, e.id_region, anio
)
SELECT
  r.nombre AS region,
  e.nombre AS embalse,
  c.anio,
  c.dias_con_aporte,
  d.total_dias,
  ROUND((c.dias_con_aporte::NUMERIC / d.total_dias) * 100, 2) AS porcentaje_completitud
FROM conteo_aportes c
JOIN dias_por_anio d ON c.anio = d.anio
JOIN corregido.embalses e ON c.id_embalse = e.id
JOIN corregido.regiones r ON e.id_region = r.id
ORDER BY r.nombre, e.nombre, c.anio;

```

## QUERY PLAN

```

Incremental Sort (cost=1740.28..2463.63 rows=71 width=154) (actual time=27.537..27.674 rows=88 loops=1)
  Sort Key: r.nombre, e.nombre, (EXTRACT(year FROM ah.fecha))
  Presorted Key: r.nombre
  Full-sort Groups: 3 Sort Method: quicksort Average Memory: 27kB Peak Memory: 27kB
-> Nested Loop (cost=1595.81..2461.36 rows=71 width=154) (actual time=27.009..27.516 rows=88 loops=1)
  Join Filter: (r.id = e.id_region)
  Rows Removed by Join Filter: 440
-> Index Scan using idx_regiones_nombre on regiones r (cost=0.13..12.22 rows=6 width=62) (actual time=0.058..0.06...)

```

## SALIDA DE CONSULTA

Antioquia	BOCA	2023	365	365	100.00
Antioquia	BOCA	2024	366	366	100.00
Antioquia	CLLR	2023	365	365	100.00
Antioquia	CLLR	2024	366	366	100.00

Antioquia	DESV	2023	1978	365	541.92
Antioquia	DESV	2024	1909	366	521.58
Antioquia	ESCM	2023	122	365	33.42
Antioquia	ESCM	2024	366	366	100.00
Antioquia	ITUA	2023	365	365	100.00
Antioquia	ITUA	2024	366	366	100.00
Antioquia	MIRF	2023	365	365	100.00
Antioquia	MIRF	2024	366	366	100.00
Antioquia	PENO	2023	365	365	100.00
Antioquia	PENO	2024	366	366	100.00
Antioquia	PLAY	2023	365	365	100.00
Antioquia	PLAY	2024	366	366	100.00
Antioquia	PP-2	2023	365	365	100.00
Antioquia	PP-2	2024	366	366	100.00
Antioquia	PP-3	2023	365	365	100.00
Antioquia	PP-3	2024	366	366	100.00
Antioquia	PUNC	2023	365	365	100.00
Antioquia	PUNC	2024	366	366	100.00
Antioquia	QUBR	2023	365	365	100.00
Antioquia	QUBR	2024	366	366	100.00
Antioquia	RGR2	2023	365	365	100.00
Antioquia	RGR2	2024	366	366	100.00
Antioquia	SLOR	2023	365	365	100.00
Antioquia	SLOR	2024	366	366	100.00
Antioquia	SMIG	2023	364	365	99.73
Antioquia	SMIG	2024	366	366	100.00

Antioquia	TRON	2023	365	365	100.00
Antioquia	TRON	2024	366	366	100.00
Caldas	CAME	2023	244	365	66.85
Caldas	CAME	2023	244	365	66.85
Caldas	CAME	2024	732	366	200.00
Caldas	CAME	2024	732	366	200.00
Caldas	DESV	2023	1978	365	541.92
Caldas	DESV	2023	1978	365	541.92
Caldas	DESV	2024	1909	366	521.58
Caldas	DESV	2024	1909	366	521.58
Caldas	ESME	2023	487	365	133.42
Caldas	ESME	2024	732	366	200.00
Caldas	ESTR	2023	122	365	33.42
Caldas	ESTR	2024	365	366	99.73
Caldas	PTEH	2023	365	365	100.00
Caldas	PTEH	2024	366	366	100.00
Caldas	SANF	2023	122	365	33.42
Caldas	SANF	2024	366	366	100.00
Caribe URR1		2023	365	365	100.00
Caribe URR1		2024	366	366	100.00
CentroALIC		2023	355	365	97.26
CentroALIC		2024	322	366	87.98
CentroAMOY		2023	365	365	100.00
CentroAMOY		2024	366	366	100.00
CentroBETA		2023	365	365	100.00
CentroBETA		2024	366	366	100.00

CentroCUCU	2023	365	365	100.00	
CentroCUCU	2024	366	366	100.00	
CentroDESV	2023	1978	365	541.92	
CentroDESV	2024	1909	366	521.58	
CentroEMBA	2023	968	365	265.21	
CentroEMBA	2024	719	366	196.45	
CentroQUIM	2023	365	365	100.00	
CentroQUIM	2024	366	366	100.00	
CentroSOGA	2023	365	365	100.00	
CentroSOGA	2024	366	366	100.00	
Oriente	DESV	2023	1978	365	541.92
Oriente	DESV	2023	1978	365	541.92
Oriente	DESV	2024	1909	366	521.58
Oriente	DESV	2024	1909	366	521.58
Oriente	EMBA	2023	968	365	265.21
Oriente	EMBA	2023	968	365	265.21
Oriente	EMBA	2024	719	366	196.45
Oriente	EMBA	2024	719	366	196.45
Oriente	ESME	2023	487	365	133.42
Oriente	ESME	2024	732	366	200.00
Oriente	GUAV	2023	122	365	33.42
Oriente	GUAV	2024	366	366	100.00
Valle	ALTO	2023	365	365	100.00
Valle	ALTO	2024	366	366	100.00
Valle	BAJO	2023	365	365	100.00
Valle	BAJO	2024	366	366	100.00



Valle	CAL1	2023	365	365	100.00
Valle	CAL1	2024	366	366	100.00
Valle	FLR2	2023	364	365	99.73
Valle	FLR2	2024	365	366	99.73
Valle	SALV	2023	365	365	100.00
Valle	SALV	2024	366	366	100.00

#### Etapa 4: Diagnóstico de niveles mínimos de aporte hídrico– 30%

Antes de índices

```
EXPLAIN ANALYZE
WITH aportes_2024 AS (
  SELECT
    ah.id_embalse,
    ah.aporte_hidrico
  FROM corregido.aportes_hidricos ah
  WHERE EXTRACT(YEAR FROM ah.fecha) = 2024
),
valores_extremos AS (
  SELECT
    id_embalse,
    MIN(aporte_hidrico) AS valor_minimo,
    MAX(aporte_hidrico) AS valor_maximo
  FROM aportes_2024
  GROUP BY id_embalse
)
SELECT
  e.nombre AS embalse,
  v.valor_maximo,
  v.valor_minimo,
  CASE
    WHEN v.valor_maximo = 0 THEN NULL -- Para evitar división por cero
    ELSE ((v.valor_maximo - v.valor_minimo) / v.valor_maximo) * 100
  END AS porcentaje_reduccion
FROM valores_extremos v
JOIN corregido.embalses e ON v.id_embalse = e.id
ORDER BY porcentaje_reduccion DESC;
```

##### QUERY PLAN

Sort (cost=1195.35..1195.46 rows=44 width=44) (actual time=13.162..13.166 rows=44 loops=1)

Sort Key: (CASE WHEN (v.valor\_maximo = '0'::double precision) THEN NULL::double precision ELSE (((v.valor\_maximo - v....

Sort Method: quicksort Memory: 27kB

-> Hash Join (cost=1166.53..1194.15 rows=44 width=44) (actual time=13.123..13.138 rows=44 loops=1)

Hash Cond: (e.id = v.id\_embalse)

Después de índices

```

EXPLAIN ANALYZE
WITH aportes_2024 AS (
  SELECT
    ah.id_embalse,
    ah.aporte_hidrico
  FROM corregido.aportes_hidricos ah
  WHERE EXTRACT(YEAR FROM ah.fecha) = 2024
),
valores_extremos AS (
  SELECT
    id_embalse,
    MIN(aporte_hidrico) AS valor_minimo,
    MAX(aporte_hidrico) AS valor_maximo
  FROM aportes_2024
  GROUP BY id_embalse
)

```

#### QUERY PLAN

```

Sort (cost=1169.74..1169.85 rows=44 width=44) (actual time=11.255..11.259 rows=44 loops=1)

  Sort Key: (CASE WHEN (v.valor_maximo = '0'::double precision) THEN NULL::double precision ELSE (((v.valor_maximo - v...

  Sort Method: quicksort  Memory: 27kB

-> Hash Join (cost=1166.53..1168.54 rows=44 width=44) (actual time=11.210..11.222 rows=44 loops=1)

  Hash Cond: (e.id = v.id_embalse)

-> Seq Scan on embalses e (cost=0.00..1.44 rows=44 width=24) (actual time=0.004..0.007 rows=44 loops=1)

-> Hash (cost=1165.98..1165.98 rows=44 width=20) (actual time=11.193..11.194 rows=44 loops=1)

```

#### SALIDA DE CONSULTA

EMBA	22580100	1100	99.99512845381552
EMBA	22580100	1100	99.99512845381552
EMBA	22580100	1100	99.99512845381552
DESV	8153600	500	99.99386773940346
DESV	8153600	500	99.99386773940346
DESV	8153600	500	99.99386773940346
DESV	8153600	500	99.99386773940346
DESV	8153600	500	99.99386773940346
DESV	8153600	500	99.99386773940346
ALIC	65239800	26900	99.95876750082006
ESME	217662600	397100	99.81756167573116
ESME	217662600	397100	99.81756167573116

CAL1	1650300	4900	99.70308428770527
MIRF	5211300	19000	99.63540767179015
SANF	12609000	72900	99.42184154175588
BOCA	10983600	130200	98.81459630722168
PP-3	4772900	69400	98.54595738439941
GUAV	88338600	1584200	98.20667296063102
ESTR	1866100	42700	97.71180536948717
SOGA	63621400	1886900	97.03417403578042
BETA	7069700	218400	96.91076000396056
PLAY	26584800	909400	96.57924829225722
PP-2	15300100	530300	96.53400958163672
CLLR	25508400	1085900	95.74297094290507
URR1	14302900	639700	95.52748044102944
ITUA	97578500	4475400	95.4135388430853
RGR2	42937300	2199900	94.87648268521774
QUBR	328700	16900	94.85853361728019
CUCU	3303100	172700	94.77157821440466
SLOR	40150400	2137800	94.67552004463218
TRON	25956600	1441200	94.44765493169368
QUIM	43267100	2440100	94.36038005782684
CAME	6137800	355200	94.21291016325067
CAME	6137800	355200	94.21291016325067
PTEH	14533700	986800	93.21026304382228
SALV	17111100	1295400	92.42947560355559
ALTO	14692700	1138800	92.24921219381054
PUNC	20078100	1719300	91.43693875416498

BAJO	1134800	108500	90.4388438491364
FLR2	439000	42400	90.34168564920273
PENO	16096800	1928700	88.018115401819
AMOY	6045200	735800	87.82835969033283
ESCM	28744800	3922300	86.35474938075757
SMIG	3909800	673900	82.76382423653384