

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.m

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 SELECT * FROM EvaluationSummary ORDER BY evaluation_id;
2 SELECT * FROM Feedback ORDER BY feedback_id LIMIT 20;
3 SELECT * FROM HIER ORDER BY parent_id, child_id;
4 SELECT * FROM EvaluationSummary_AUDIT ORDER BY changed_at LIMIT 10;
5
6
7
```

Data Output Messages Notifications

Query returned successfully in 150 msec.

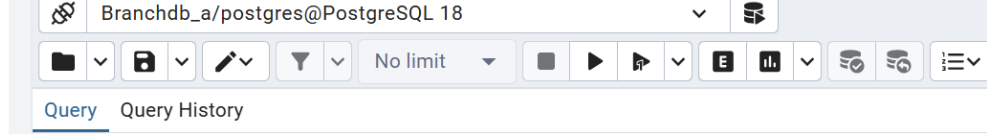
Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 SELECT
2   (SELECT count(*) FROM EvaluationSummary) AS cnt_evaluation_summary,
3   (SELECT count(*) FROM Feedback AS cnt_feedback,
4   (SELECT count(*) FROM HIER) AS cnt_hier,
5   (SELECT count(*) FROM TRIPLE) AS cnt_triple,
6   (SELECT count(*) FROM BUSINESS_LIMITS) AS cnt_business_limits,
7   (SELECT count(*) FROM EvaluationSummary_AUDIT) AS cnt_audit;
8
9
```

b\_b;



```
1  WITH RECURSIVE rollup(child_id, root_id, depth) AS (  
2      SELECT child_id, parent_id AS root_id, 1 AS depth FROM HIER  
3      UNION  
4      SELECT h.child_id, r.root_id, r.depth + 1  
5      FROM HIER h  
6      JOIN rollup r ON h.parent_id = r.child_id  
7  )  
8  SELECT DISTINCT child_id, root_id, depth  
9  FROM rollup  
10 ORDER BY root_id, depth;  
11  
12
```

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.m

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 SELECT * FROM EvaluationSummary ORDER BY evaluation_id;
2 SELECT * FROM Feedback ORDER BY feedback_id LIMIT 20;
3 SELECT * FROM HIER ORDER BY parent_id, child_id;
4 SELECT * FROM EvaluationSummary_AUDIT ORDER BY changed_at LIMIT 10;
5
6
7
```

Data Output Messages Notifications

Query returned successfully in 150 msec.

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 SELECT
2 (SELECT count(*) FROM EvaluationSummary) AS cnt_evaluation_summary,
3 (SELECT count(*) FROM Feedback AS cnt_feedback,
4 (SELECT count(*) FROM HIER) AS cnt_hier,
5 (SELECT count(*) FROM TRIPLE) AS cnt_triple,
6 (SELECT count(*) FROM BUSINESS_LIMITS) AS cnt_business_limits,
7 (SELECT count(*) FROM EvaluationSummary_AUDIT) AS cnt_audit;
8
9
```

b\_b;

Branchdb\_b/postg... x Branch\_ab/postgr... x Branchdb\_a/postgres@PostgreSQL 18\* x Read.me/p

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 WITH RECURSIVE rollup(child_id, root_id, depth) AS (  
2     SELECT child_id, parent_id AS root_id, 1 AS depth FROM HIER  
3     UNION  
4     SELECT h.child_id, r.root_id, r.depth + 1  
5     FROM HIER h  
6     JOIN rollup r ON h.parent_id = r.child_id  
7 )  
8 SELECT DISTINCT child_id, root_id, depth  
9 FROM rollup  
10 ORDER BY root_id, depth;  
11  
12
```

Edit View Window Help

Branchdb\_b/postg... x Branch\_ab/postgr... x Branchdb\_a/postgres@PostgreSQL 18\* x Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 CREATE OR REPLACE FUNCTION trg_recompute_totals_statement() RETURNS TRIGGER LANGUAGE plpgsql AS  
2 DECLARE  
3     eval_id INTEGER;  
4     bef_total INTEGER;  
5     aft_total INTEGER;  
6 BEGIN
```

Branchdb\_b/postg... x Branch\_ab/postgr... x Branchdb\_a/postgres@PostgreSQL 18\* x Read.i

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 CREATE TABLE IF NOT EXISTS EvaluationSummary_AUDIT (  
2     audit_id SERIAL PRIMARY KEY,  
3     key_col VARCHAR(64),  
4     bef_total INTEGER,  
5     aft_total INTEGER,  
6     changed_at TIMESTAMP DEFAULT now()  
7 );  
8
```

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 WITH one_eval AS (  
2     SELECT evaluation_id FROM EvaluationSummary ORDER BY evaluation_id LIMIT 1  
3 )  
4 INSERT INTO Feedback (evaluation_id, studentid, rating, comment)  
5 SELECT evaluation_id, 1001, 4, 'Passing feedback A' FROM one_eval  
6 ON CONFLICT DO NOTHING;  
7  
8 WITH one_eval AS (  
9     SELECT evaluation_id FROM EvaluationSummary ORDER BY evaluation_id DESC LIMIT 1  
10 )  
11 INSERT INTO Feedback (evaluation_id, studentid, rating, comment)  
12 SELECT evaluation_id, 1002, 5, 'Passing feedback B' FROM one_eval  
13 ON CONFLICT DO NOTHING;  
14  
15 DO $$  
16 BEGIN
```

File View Window Help

er

Foreign Data Wrappers  
Languages  
Publications  
Schemas  
Subscriptions  
CREATE DATABASE branchdb\_b;  
Node\_A  
Casts  
Catalogs  
Event Triggers  
Extensions  
Foreign Data Wrappers  
Languages  
Publications

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 END;  
2  
3 BEGIN  
4     INSERT INTO EvaluationSummary (summary_text, total_rating)  
5     VALUES ('Bad total', -5); -- will violate es_total_nonneg_check  
6     EXCEPTION WHEN OTHERS THEN  
7         RAISE NOTICE 'Expected EvaluationSummary failure (negative total): %', SQLERRM;  
8     END;  
9 END;  
10 $$;
```

Foreign Data Wrappers  
Languages  
Publications  
Schemas  
Subscriptions  
CREATE DATABASE branchdb\_b;  
le\_A  
Casts  
Catalogs  
Event Triggers  
Extensions  
Foreign Data Wrappers  
Languages  
Publications

Branchdb\_a/postgres@PostgreSQL 18

```
1 DO $$
2 BEGIN
3     BEGIN
4         INSERT INTO EvaluationSummary (summary_text, total_rating)
5         VALUES (repeat('X', 2000), 0); -- will violate es_summary_length_check
6     EXCEPTION WHEN OTHERS THEN
7         RAISE NOTICE 'Expected EvaluationSummary failure (too long summary): %', SQLERRM;
```

Branchdb\_b/postg... x Branch\_ab/postgr... x Branchdb\_a/postgres@PostgreSQL 18\* x Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

```
1 FOR eval_id IN SELECT DISTINCT evaluation_id FROM Feedback
2 LOOP
3     SELECT COALESCE(total_rating,0) INTO bef_total FROM EvaluationSummary WHERE evaluation_id = eval_id;
4     SELECT COALESCE(SUM(rating),0) INTO aft_total FROM Feedback WHERE evaluation_id = eval_id;
5
6     IF EXISTS (SELECT 1 FROM EvaluationSummary WHERE evaluation_id = eval_id) THEN
7         UPDATE EvaluationSummary SET total_rating = aft_total WHERE evaluation_id = eval_id;
8     ELSE
9         INSERT INTO EvaluationSummary (evaluation_id, summary_text, total_rating)
10        VALUES (eval_id, 'auto-created summary', aft_total);
11    END IF;
```

File Edit View Window Help

Branchdb\_b/postg... Branch\_ab/postgr... Branchdb\_a/postgres@PostgreSQL 18\* Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

No limit

Query Query History

```
1 CREATE OR REPLACE FUNCTION trg_recompute_totals_statement() RETURNS TRIGGER LANGUAGE plpgsql AS
2 DECLARE
3     eval_id INTEGER;
4     bef_total INTEGER;
5     aft_total INTEGER;
6 BEGIN
```

Branchdb\_b/postg... Branch\_ab/postgr... Branchdb\_a/postgres@PostgreSQL 18\* Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

No limit

Query Query History

```
1 CREATE TABLE IF NOT EXISTS EvaluationSummary_AUDIT (
2     audit_id SERIAL PRIMARY KEY,
3     key_col VARCHAR(64),
4     bef_total INTEGER,
5     aft_total INTEGER,
6     changed_at TIMESTAMP DEFAULT now()
7 );
8
```

File Edit View Window Help

Branchdb\_b/postg... Branch\_ab/postgr... Branchdb\_a/postgres@PostgreSQL 18\* Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

No limit

Query Query History

```
1 DELETE FROM TRIPLE
2 WHERE s IN ('Dog', 'Mammal', 'Poodle', 'Beagle', 'Car', 'Vehicle', 'Bird', 'Sparrow');
```

Branchdb\_b/postg... Branch\_ab/postgr... Branchdb\_a/postgres@PostgreSQL 18\* Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

No limit

Query Query History

```
1 WITH RECURSIVE isa(root, typ, depth) AS (
2     SELECT s AS root, o AS typ, 1 AS depth FROM TRIPLE WHERE p = 'isA'
3     UNION
4     SELECT isa.root, t.o, isa.depth + 1
5     FROM TRIPLE t
6     JOIN isa ON t.s = isa.typ
7     WHERE t.p = 'isA'
8 )
9 SELECT root, typ AS inferred_type, MIN(depth) AS depth
10 FROM isa
11 GROUP BY root, typ
12 ORDER BY root, depth
13 LIMIT 10;
```

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 WITH RECURSIVE isa(root, typ, depth) AS (  
2   SELECT s AS root, o AS typ, 1 AS depth FROM TRIPLE WHERE p = 'isA'  
3   UNION  
4   SELECT isa.root, t.o, isa.depth + 1  
5   FROM TRIPLE t  
6   JOIN isa ON t.s = isa.typ  
7   WHERE t.p = 'isA'  
8 )
```

Window Help

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.m

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 UPDATE Feedback  
2 SET rating = 4  
3 WHERE studentid = 1001;
```

Foreign Data Wrappers  
Languages  
Publications  
Schemas  
Subscriptions  
CREATE DATABASE branchdb\_b;  
Node\_A  
Casts  
Catalogs

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 INTO feedback (evaluation_id, studentid, rating, comment)  
2  
3 ECT evaluation_id FROM EvaluationSummary ORDER BY evaluation_id LIMIT 1), 2001, 5, 'Audit test 1'),  
4 ECT evaluation_id FROM EvaluationSummary ORDER BY evaluation_id LIMIT 1), 2002, 4, 'Audit test 2');
```



File Edit View Window Help

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 DROP TRIGGER IF EXISTS feedback_recompute_totals ON Feedback;
2 CREATE TRIGGER feedback_recompute_totals
3 AFTER INSERT OR UPDATE OR DELETE ON Feedback
4 FOR EACH STATEMENT
5 EXECUTE FUNCTION trg_recompute_totals_statement();
```

Tools Edit View Window Help

Explorer

- Foreign Data Wrappers
- Languages
- Publications
- Schemas
- Subscriptions
- CREATE DATABASE branchdb\_b;
- Node\_A
- Casts
- Catalogs
- Event Triggers
- Extensions

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 FOR eval_id IN SELECT DISTINCT evaluation_id FROM Feedback
2 LOOP
3     SELECT COALESCE(total_rating,0) INTO bef_total FROM EvaluationSummary WHERE evaluation_id = eval_id;
4     SELECT COALESCE(SUM(rating),0) INTO aft_total FROM Feedback WHERE evaluation_id = eval_id;
5
6     IF EXISTS (SELECT 1 FROM EvaluationSummary WHERE evaluation_id = eval_id) THEN
7         UPDATE EvaluationSummary SET total_rating = aft_total WHERE evaluation_id = eval_id;
8     ELSE
9         INSERT INTO EvaluationSummary (evaluation_id, summary_text, total_rating)
10        VALUES (eval_id, 'auto-created summary', aft_total);
11    END IF;
```

Tools Edit View Window Help

Explorer

- Foreign Data Wrappers
- Languages
- Publications
- Schemas
- Subscriptions
- CREATE DATABASE branchdb\_b;
- Node\_A
- Casts
- Catalogs
- Event Triggers
- Extensions
- Foreign Data Wrappers
- Languages
- Publications

File Edit View Window Help

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 DO $$
2 BEGIN
3     UPDATE Feedback SET rating = 5 WHERE studentid = 1002;
4 EXCEPTION WHEN OTHERS THEN
5     RAISE NOTICE 'Expected business-limit failure (UPDATE): %', SQLERRM;
6 END;
7 $$;
```

Tools Edit View Window Help

Explorer

- Foreign Data Wrappers
- Languages
- Publications
- Schemas
- Subscriptions
- CREATE DATABASE branchdb\_b;
- Node\_A
- Casts
- Catalogs
- Event Triggers
- Extensions

File Edit View Window Help

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 DO $$
2 BEGIN
3     INSERT INTO Feedback (evaluation_id, studentid, rating, comment)
4     VALUES ((SELECT evaluation_id FROM EvaluationSummary ORDER BY evaluation_id LIMIT 1), 9001, 5, '
5 EXCEPTION WHEN OTHERS THEN
6     RAISE NOTICE 'Expected business-limit failure (INSERT): %', SQLERRM;
7 END;
8 $$;
```

Tools Edit View Window Help

Explorer

- Foreign Data Wrappers
- Languages
- Publications
- Schemas
- Subscriptions
- CREATE DATABASE branchdb\_b;
- Node\_A
- Casts
- Catalogs
- Event Triggers

View Window Help

Branchdb\_b/postg... Branch\_ab/postgr... Branchdb\_a/postgres@PostgreSQL 18\* Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 DROP TRIGGER IF EXISTS feedback_business_limit ON Feedback;
2 CREATE TRIGGER feedback_business_limit
3 BEFORE INSERT OR UPDATE ON Feedback
4 FOR EACH ROW
5 EXECUTE FUNCTION trg_business_limit_before_feedback();
```

Edit View Window Help

Branchdb\_b/postg... Branch\_ab/postgr... Branchdb\_a/postgres@PostgreSQL 18\* Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 will_alert := fn_should_alert(check_eval);
2 IF will_alert = 1 THEN
3 RAISE EXCEPTION 'Business limit violation for evaluation_id=%', check_eval;
4 END IF;
5
6 RETURN NEW;
7 END;
8 $$;
```

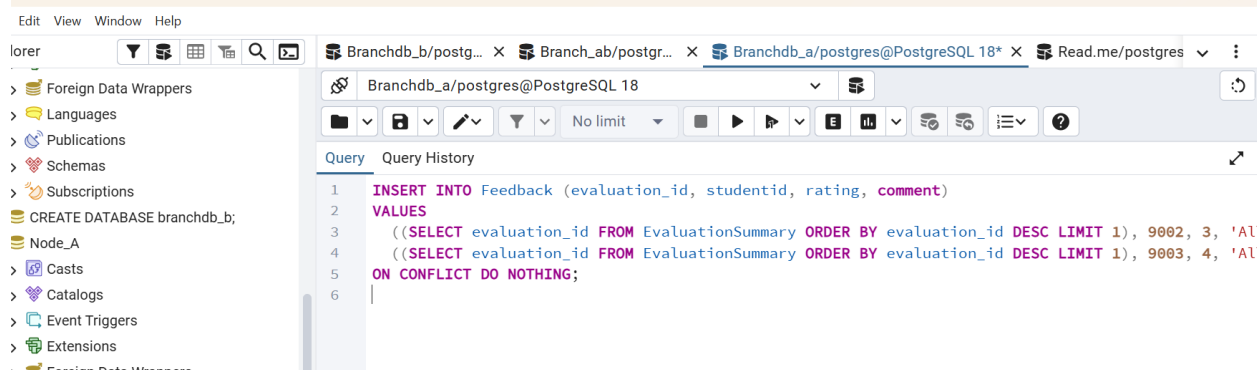
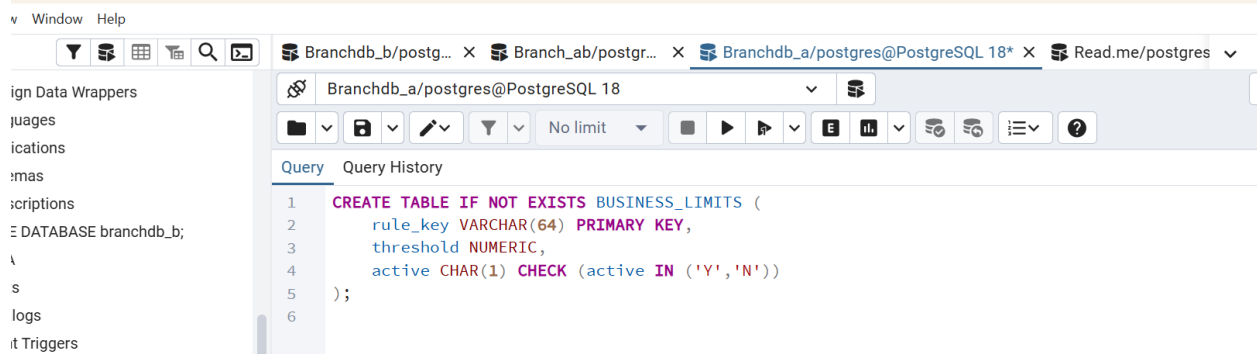
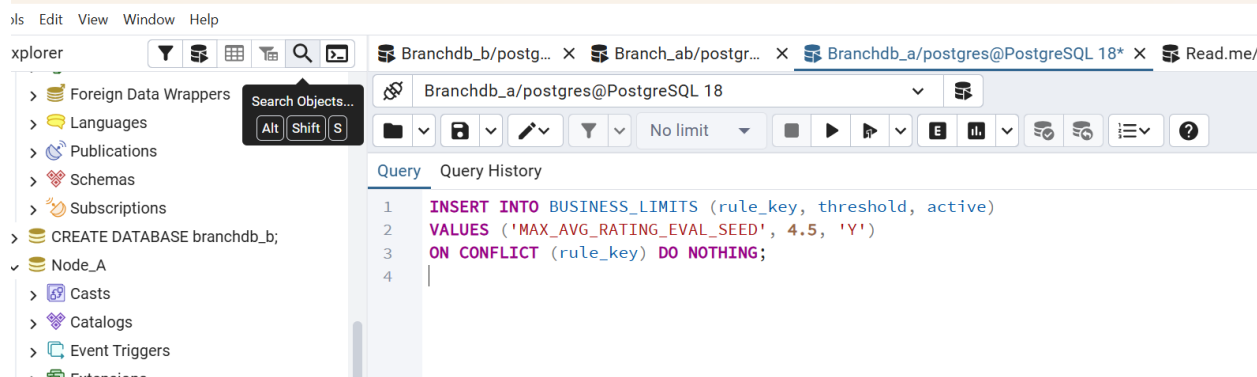
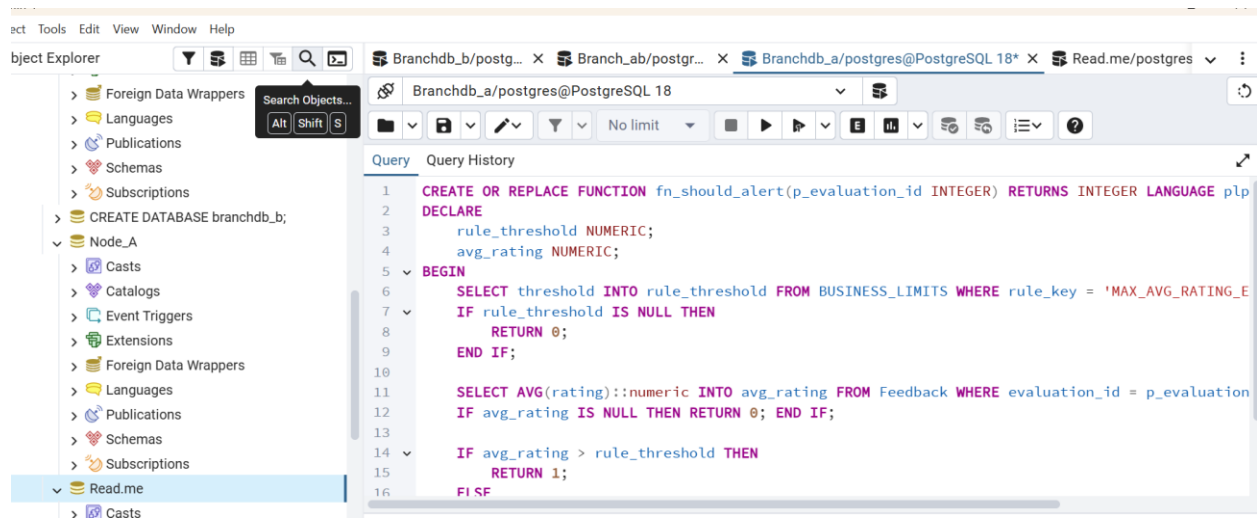
Branchdb\_b/postg... Branch\_ab/postgr... Branchdb\_a/postgres@PostgreSQL 18\* Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

Search Objects... Alt Shift S

Query Query History

```
1 CREATE OR REPLACE FUNCTION trg_business_limit_before_feedback() RETURNS TRIGGER LANGUAGE plpgsql AS
2 DECLARE
3 will_alert INTEGER;
4 check_eval INTEGER;
5 BEGIN
6 IF TG_OP = 'INSERT' OR TG_OP = 'UPDATE' THEN
7 check_eval := NEW.evaluation_id;
8 ELSE
9 RETURN NEW;
10 END IF;
11
```



Window Help

Branchdb\_b/postg... Branch\_ab/postgr... Branchdb\_a/postgres@PostgreSQL 18\* Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 SELECT * FROM EvaluationSummary ORDER BY evaluation_id;
2 SELECT * FROM Feedback ORDER BY feedback_id LIMIT 20;
3 SELECT * FROM EvaluationSummary_AUDIT ORDER BY changed_at LIMIT 10;
```

Branchdb\_b/postg... Branch\_ab/postgr... Branchdb\_a/postgres@PostgreSQL 18\* Read.m

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 SELECT
2   (SELECT count(*) FROM EvaluationSummary) AS cnt_evaluation_summary,
3   (SELECT count(*) FROM Feedback) AS cnt_feedback,
4   (SELECT count(*) FROM HIER) AS cnt_hier,
5   (SELECT count(*) FROM TRIPLE) AS cnt_triple,
6   (SELECT count(*) FROM BUSINESS_LIMITS) AS cnt_business_limits,
7   (SELECT count(*) FROM EvaluationSummary_AUDIT) AS cnt_audit;
8
9
10
```

Branchdb\_b/postg... Branch\_ab/postgr... Branchdb\_a/postgres@PostgreSQL 18\* Read.m

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 SELECT
2   (SELECT count(*) FROM EvaluationSummary) AS cnt_evaluation_summary,
3   (SELECT count(*) FROM Feedback) AS cnt_feedback,
4   (SELECT count(*) FROM HIER) AS cnt_hier,
5   (SELECT count(*) FROM TRIPLE) AS cnt_triple,
6   (SELECT count(*) FROM BUSINESS_LIMITS) AS cnt_business_limits,
7   (SELECT count(*) FROM EvaluationSummary_AUDIT) AS cnt_audit;
8
9
10
```

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.m

Wrappers

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 SELECT
2   (SELECT count(*) FROM EvaluationSummary) AS cnt_evaluation_summary,
3   (SELECT count(*) FROM Feedback) AS cnt_feedback,
4   (SELECT count(*) FROM HIER) AS cnt_hier,
5   (SELECT count(*) FROM TRIPLE) AS cnt_triple,
6   (SELECT count(*) FROM BUSINESS_LIMITS) AS cnt_business_limits,
7   (SELECT count(*) FROM EvaluationSummary_AUDIT) AS cnt_audit;
8
9
10
```

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.me/

Data Wrappers

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 DELETE FROM Feedback WHERE studentid IN (2001,2002,9002,9003,2001,2002);
2
3
```

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.me/

Data Wrappers

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 DELETE FROM Feedback WHERE studentid IN (2001,2002,9002,9003,2001,2002);
2
3
```

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 DELETE FROM HIER WHERE parent_id IN (10,14) OR child_id IN (11,12,13,14,15,16);
2
```

branchdb\_b;

ow Help

Branchdb\_b/postg... X Branch\_ab/postgr... X Branchdb\_a/postgres@PostgreSQL 18\* X Read.me/postgres

Branchdb\_a/postgres@PostgreSQL 18

Query Query History

```
1 WITH RECURSIVE rollup(child_id, root_id, depth) AS (
2   SELECT child_id, parent_id AS root_id, 1 AS depth FROM HIER
3   UNION ALL
4   SELECT h.child_id, r.root_id, r.depth + 1
5   FROM HIER h
6   JOIN rollup r ON h.parent_id = r.child_id
7 )
8 SELECT DISTINCT child_id, root_id, depth FROM rollup ORDER BY root_id, depth LIMIT 20;
```

Wrappers

is

IASE branchdb\_b;

rs

Wrappers