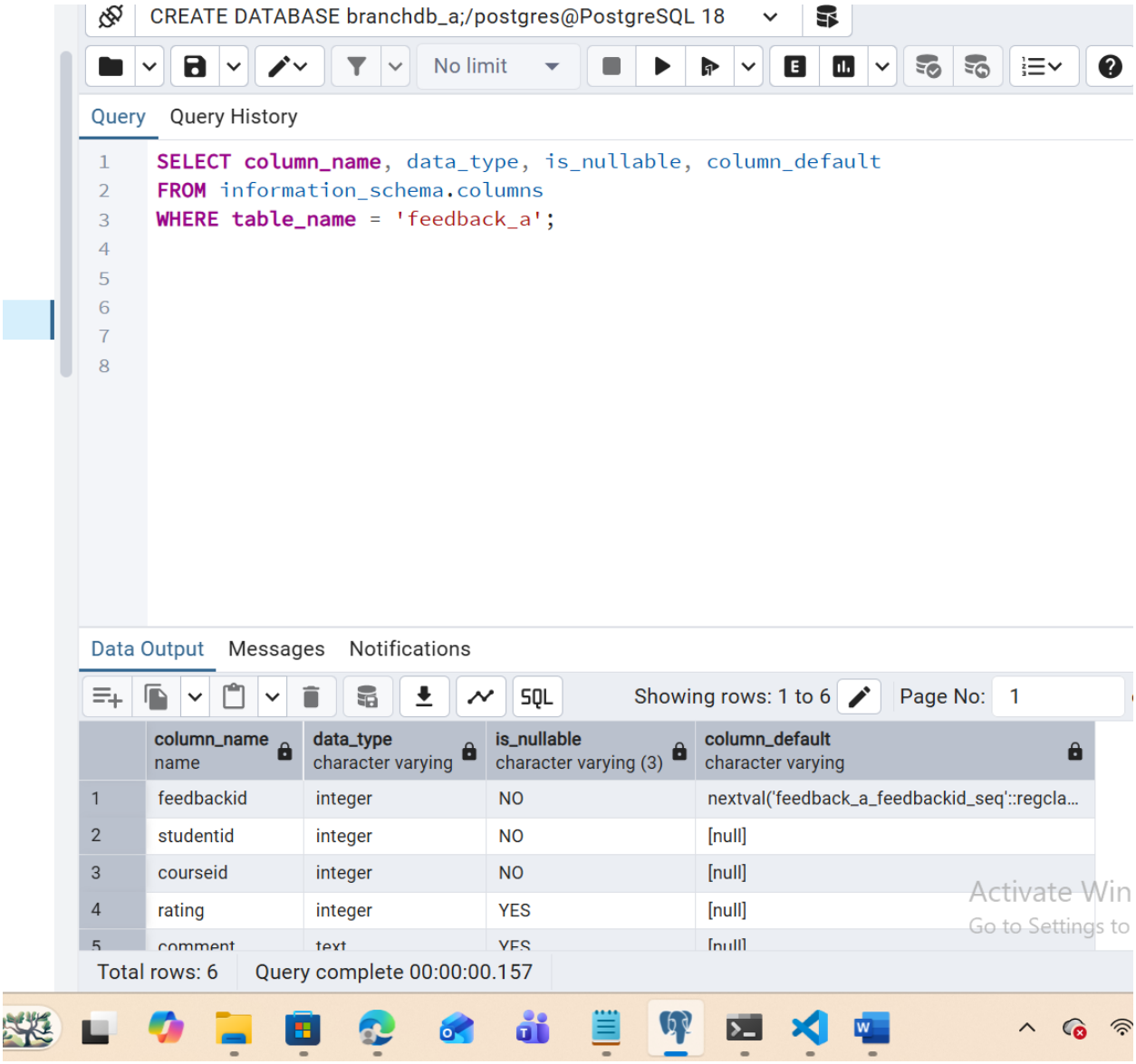
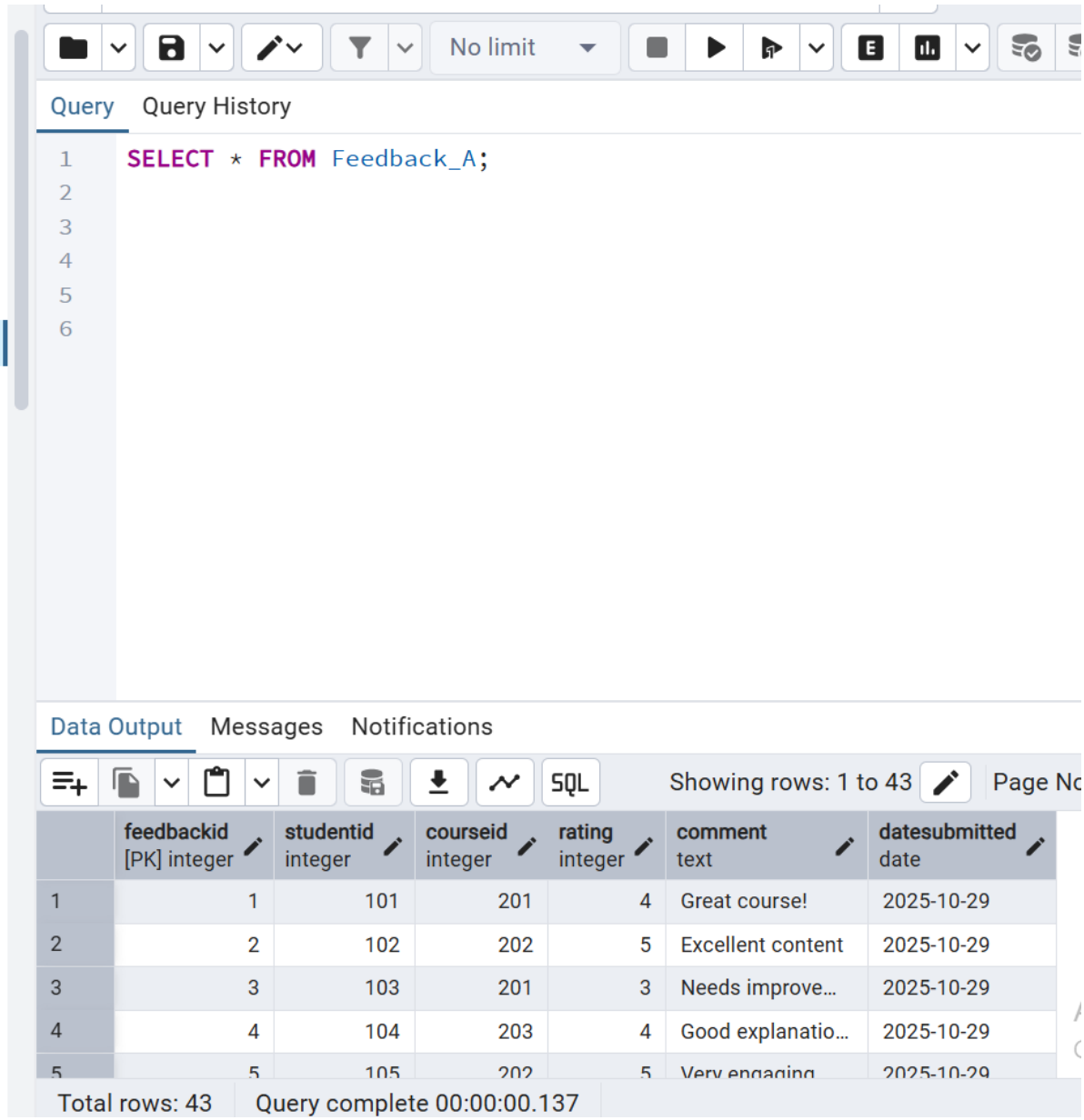
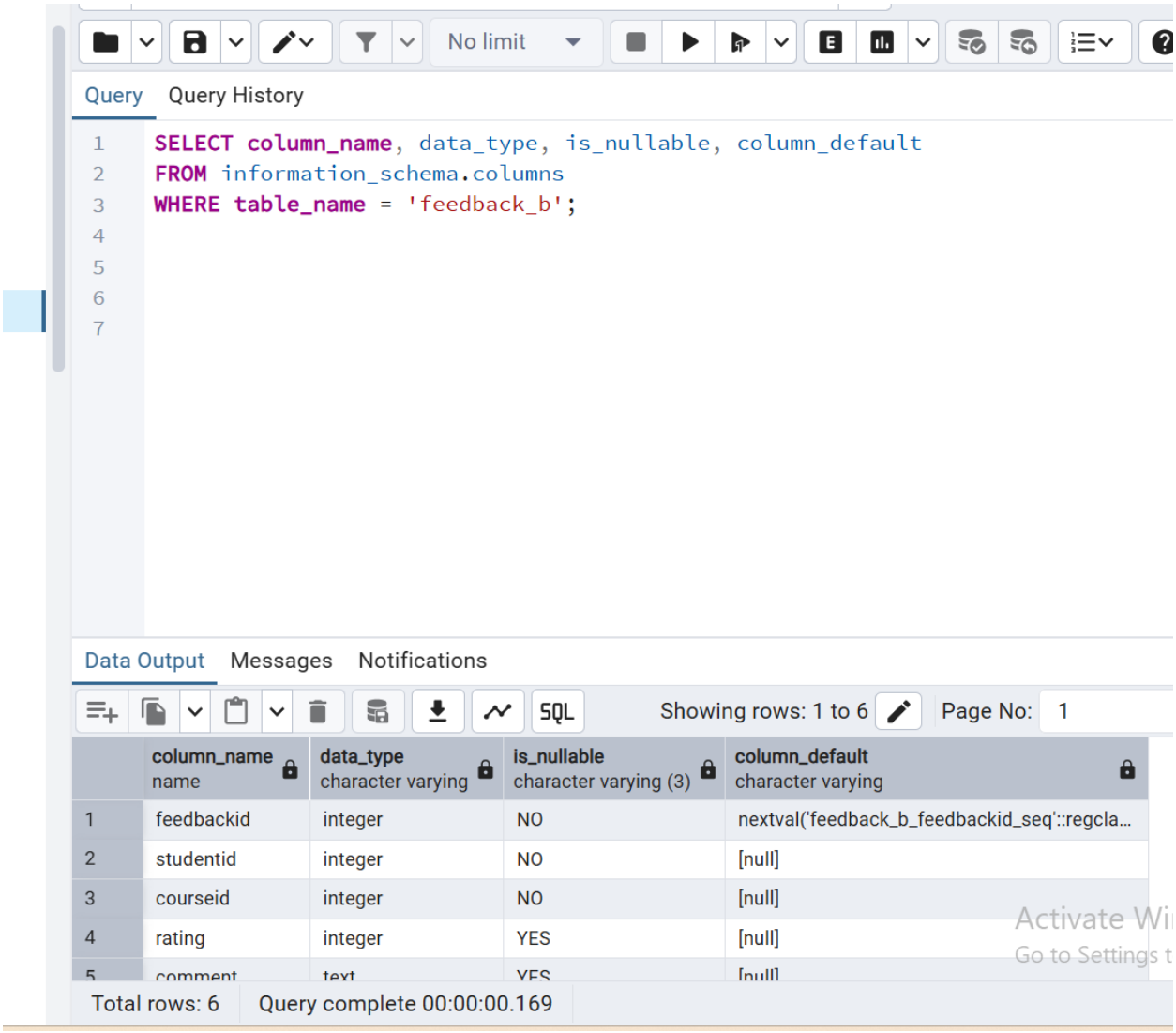
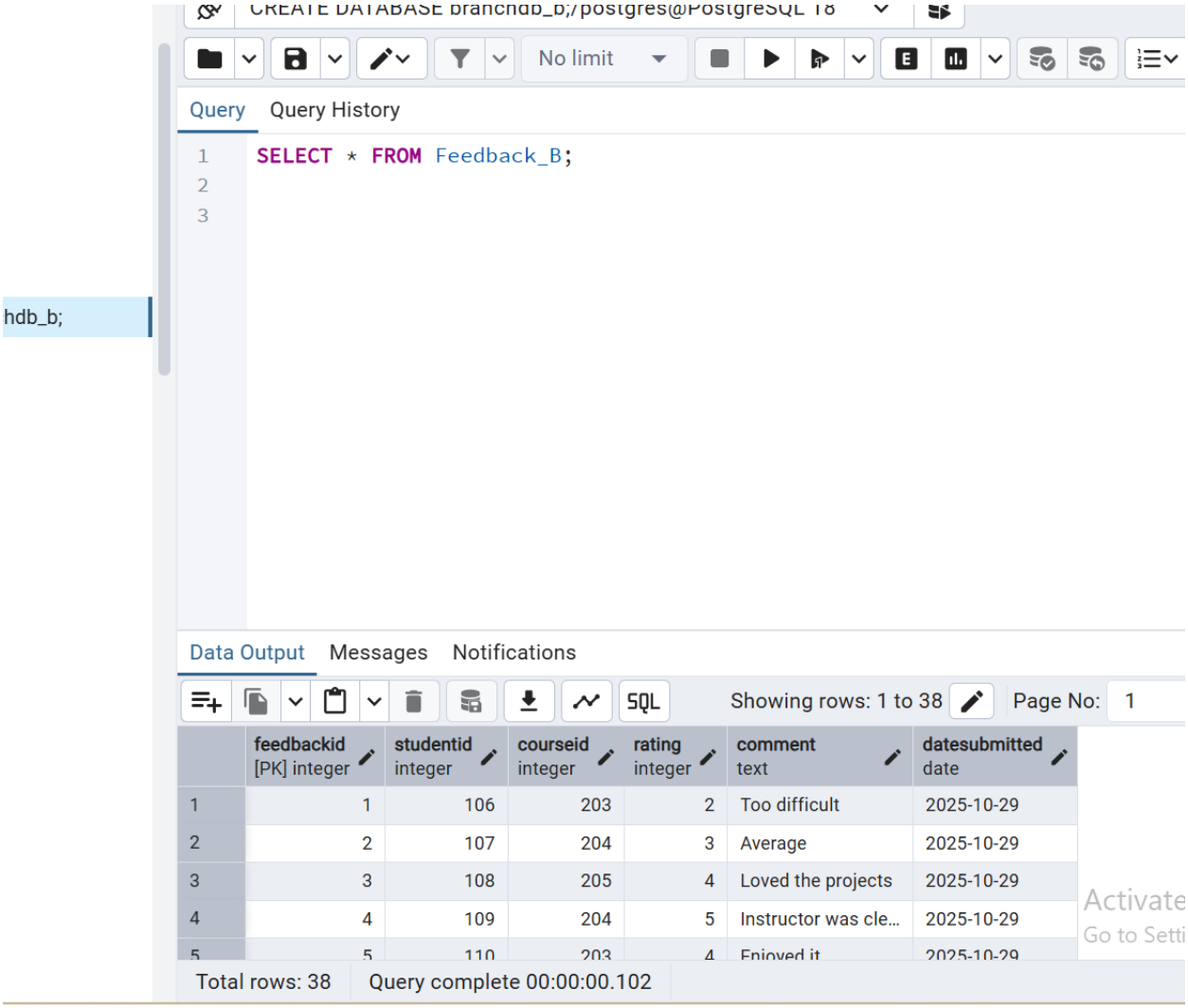
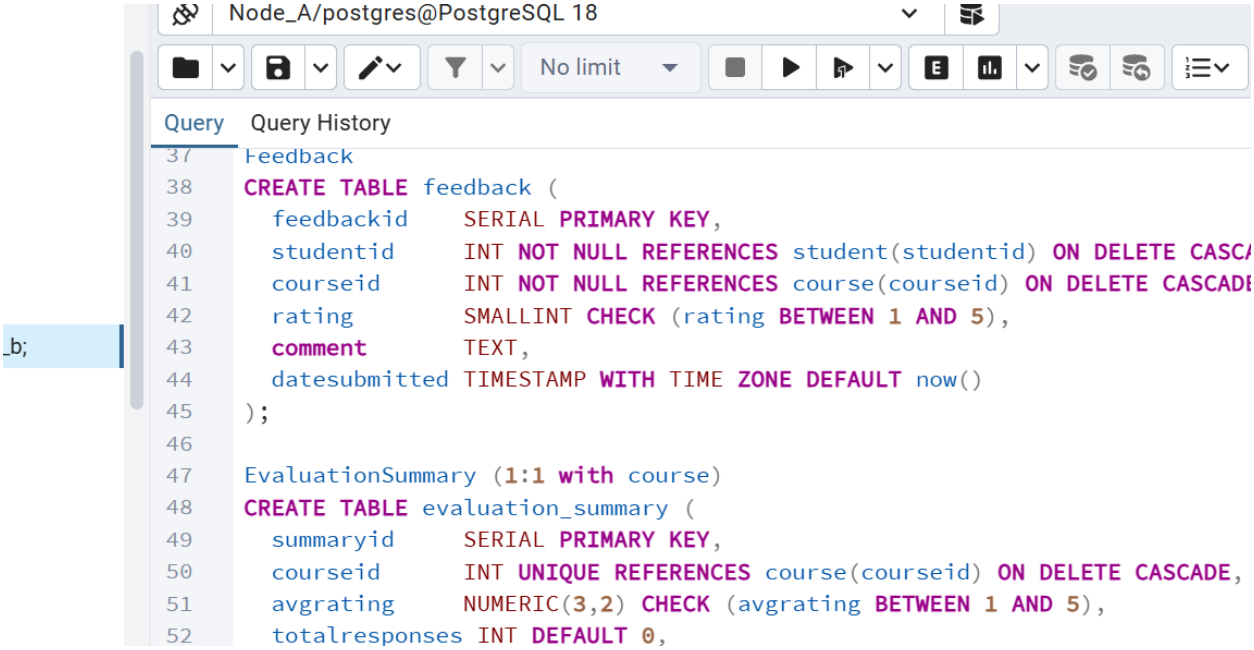
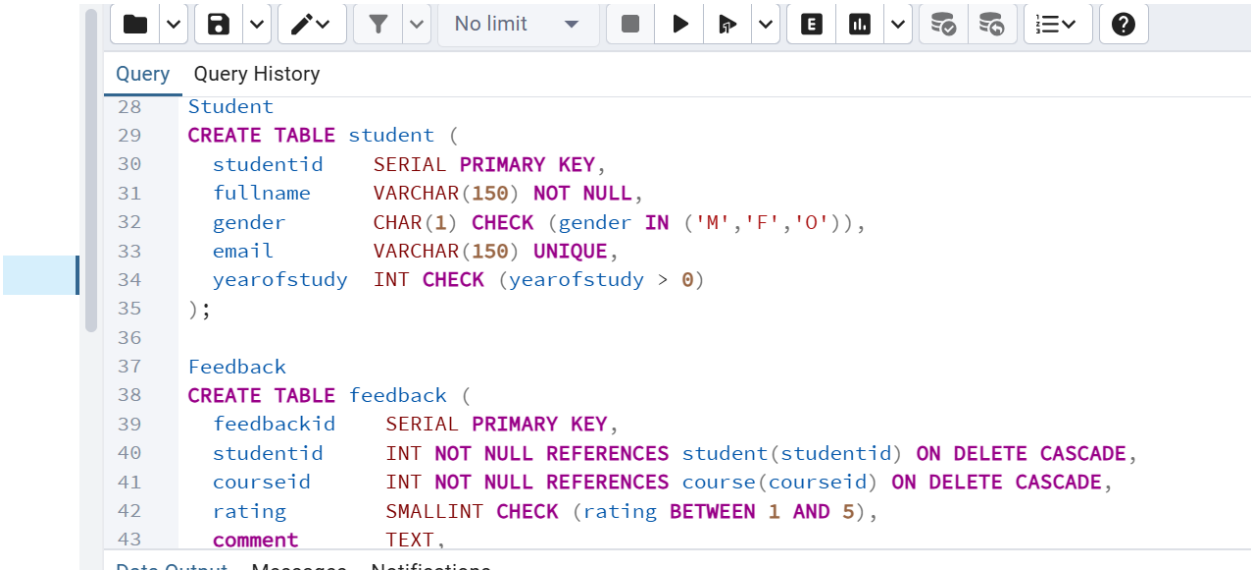

CREATE DATABASE branchdb_a;/postgres@PostgreSQL 18
No limit
Query
Query History
1 SELECT column_name, data_type, is_nullable, column_default
2 FROM information_schema.columns
3 WHERE table_name = 'feedback_a';
4
5
6
7
8
Data Output
Messages
Notifications
SQL
Showing rows: 1 to 6
Page No: 1
column_name name
data_type character varying
is_nullable character varying (3)
column_default character varying
1 feedbackid integer NO nextval('feedback_a_feedbackid_seq'::regcla...
2 studentid integer NO [null]
3 courseid integer NO [null]
4 rating integer YES [null]
5 comment text YES [null]
Total rows: 6
Query complete 00:00:00.157
Activate Win
Go to Settings to

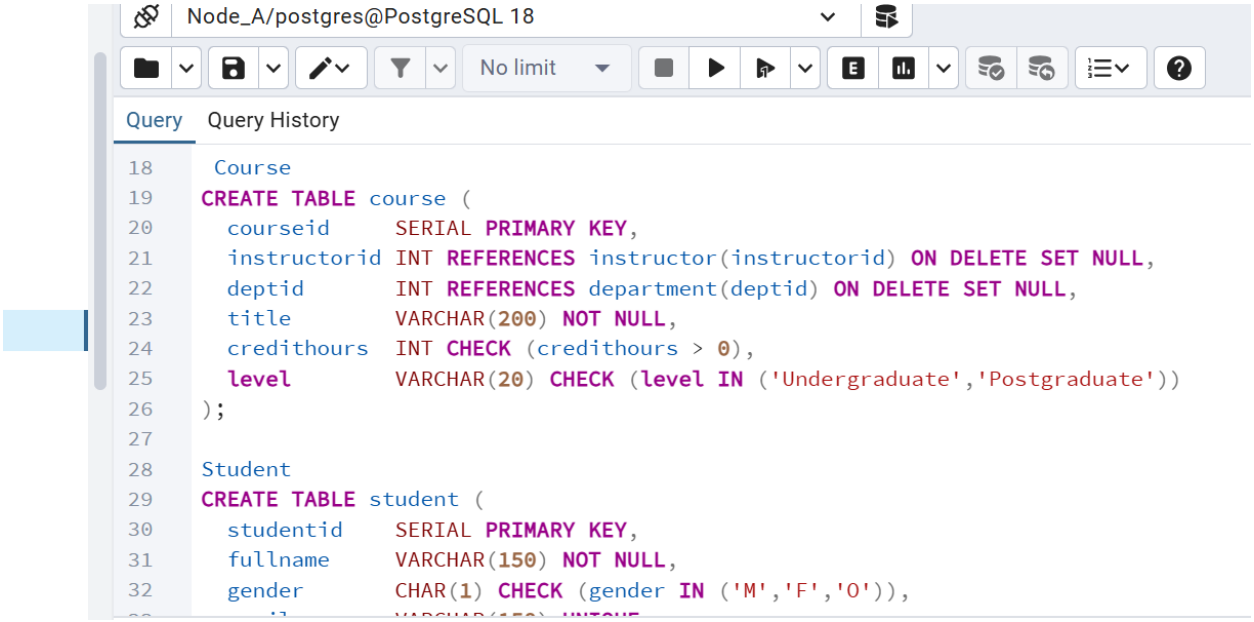

No limit
Query
Query History
1 SELECT * FROM Feedback_A;
2
3
4
5
6
Data Output
Messages
Notifications
SQL
Showing rows: 1 to 43
Page No
feedbackid [PK] integer
studentid integer
courseid integer
rating integer
comment text
datesubmitted date
1 1 101 201 4 Great course! 2025-10-29
2 2 102 202 5 Excellent content 2025-10-29
3 3 103 201 3 Needs improve... 2025-10-29
4 4 104 203 4 Good explanatio... 2025-10-29
5 5 105 202 5 Very engaging 2025-10-29
Total rows: 43
Query complete 00:00:00.137

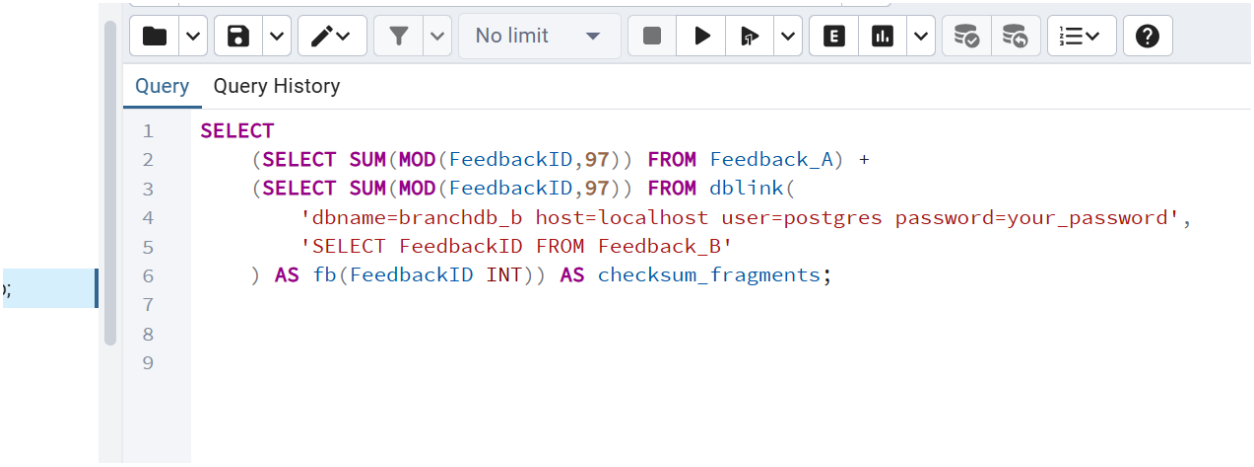

No limit
Query
Query History
1 SELECT column_name, data_type, is_nullable, column_default
2 FROM information_schema.columns
3 WHERE table_name = 'feedback_b';
4
5
6
7
Data Output
Messages
Notifications
SQL
Showing rows: 1 to 6
Page No: 1
column_name name
data_type character varying
is_nullable character varying (3)
column_default character varying
1 feedbackid integer NO nextval('feedback_b_feedbackid_seq'::regcla...
2 studentid integer NO [null]
3 courseid integer NO [null]
4 rating integer YES [null]
5 comment text YES [null]
Total rows: 6
Query complete 00:00:00.169

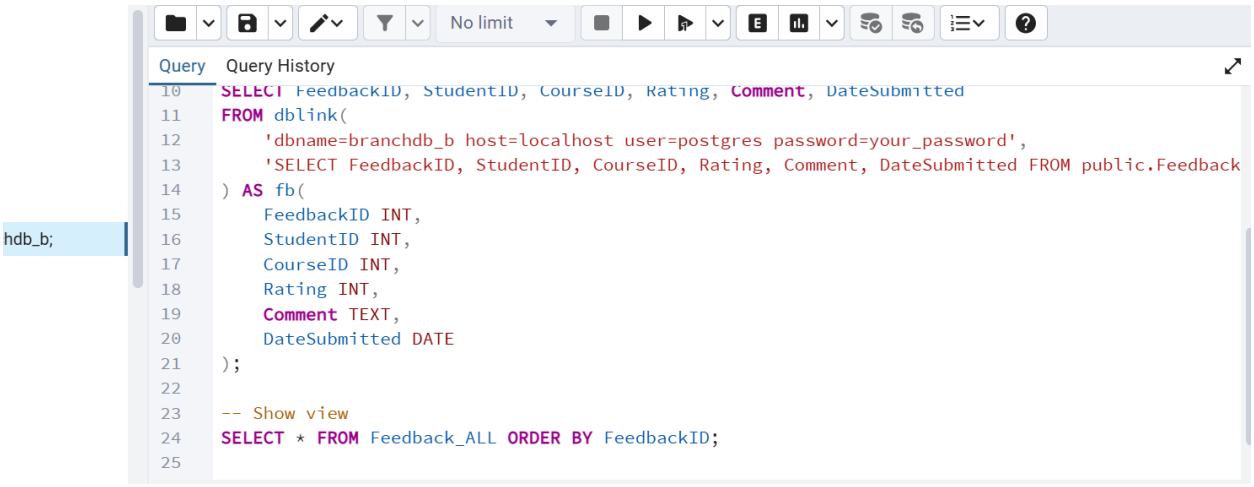

CREATE DATABASE branchdb_b;/postgres@PostgreSQL 18
No limit
Query
Query History
1 SELECT * FROM Feedback_B;
2
3
hdb_b;
Data Output
Messages
Notifications
SQL
Showing rows: 1 to 38
Page No: 1
feedbackid [PK] integer
studentid integer
courseid integer
rating integer
comment text
datesubmitted date
1 1 106 203 2 Too difficult 2025-10-29
2 2 107 204 3 Average 2025-10-29
3 3 108 205 4 Loved the projects 2025-10-29
4 4 109 204 5 Instructor was cle... 2025-10-29
5 110 203 4 Enjoyed it 2025-10-29
Activate
Go to Setti
Total rows: 38
Query complete 00:00:00.102


Node_A/postgres@PostgreSQL 18
No limit
Query
Query History
Feedback
CREATE TABLE feedback (
feedbackid SERIAL PRIMARY KEY,
studentid INT NOT NULL REFERENCES student(studentid) ON DELETE CASC
courseid INT NOT NULL REFERENCES course(courseid) ON DELETE CASCADE
rating SMALLINT CHECK (rating BETWEEN 1 AND 5),
comment TEXT,
datesubmitted TIMESTAMP WITH TIME ZONE DEFAULT now()
);
EvaluationSummary (1:1 with course)
CREATE TABLE evaluation_summary (
summaryid SERIAL PRIMARY KEY,
courseid INT UNIQUE REFERENCES course(courseid) ON DELETE CASCADE,
avgrating NUMERIC(3,2) CHECK (avgrating BETWEEN 1 AND 5),
totalresponses INT DEFAULT 0,


No limit
Query
Query History
Student
CREATE TABLE student (
studentid SERIAL PRIMARY KEY,
fullname VARCHAR(150) NOT NULL,
gender CHAR(1) CHECK (gender IN ('M','F','O')),
email VARCHAR(150) UNIQUE,
yearofstudy INT CHECK (yearofstudy > 0)
);
Feedback
CREATE TABLE feedback (
feedbackid SERIAL PRIMARY KEY,
studentid INT NOT NULL REFERENCES student(studentid) ON DELETE CASCADE,
courseid INT NOT NULL REFERENCES course(courseid) ON DELETE CASCADE,
rating SMALLINT CHECK (rating BETWEEN 1 AND 5),
comment TEXT,


Node_A/postgres@PostgreSQL 18
No limit
Query
Query History
Course
CREATE TABLE course (
courseid SERIAL PRIMARY KEY,
instructorid INT REFERENCES instructor(instructorid) ON DELETE SET NULL,
deptid INT REFERENCES department(deptid) ON DELETE SET NULL,
title VARCHAR(200) NOT NULL,
credithours INT CHECK (credithours > 0),
level VARCHAR(20) CHECK (level IN ('Undergraduate','Postgraduate'))
);
Student
CREATE TABLE student (
studentid SERIAL PRIMARY KEY,
fullname VARCHAR(150) NOT NULL,
gender CHAR(1) CHECK (gender IN ('M','F','O')),


No limit
Query
Query History
SELECT
(SELECT SUM(MOD(FeedbackID,97)) FROM Feedback_A) +
(SELECT SUM(MOD(FeedbackID,97)) FROM dblink(
'dbname=branchdb_b host=localhost user=postgres password=your_password',
'SELECT FeedbackID FROM Feedback_B'
) AS fb(FeedbackID INT)) AS checksum_fragments;


No limit
Query
Query History
SELECT FeedbackID, StudentID, CourseID, Rating, Comment, DateSubmitted
FROM dblink(
'dbname=branchdb_b host=localhost user=postgres password=your_password',
'SELECT FeedbackID, StudentID, CourseID, Rating, Comment, DateSubmitted FROM public.Feedback
) AS fb(
FeedbackID INT,
StudentID INT,
CourseID INT,
Rating INT,
Comment TEXT,
DateSubmitted DATE
);
-- Show view
SELECT * FROM Feedback_ALL ORDER BY FeedbackID;

No limit

Query | Query History

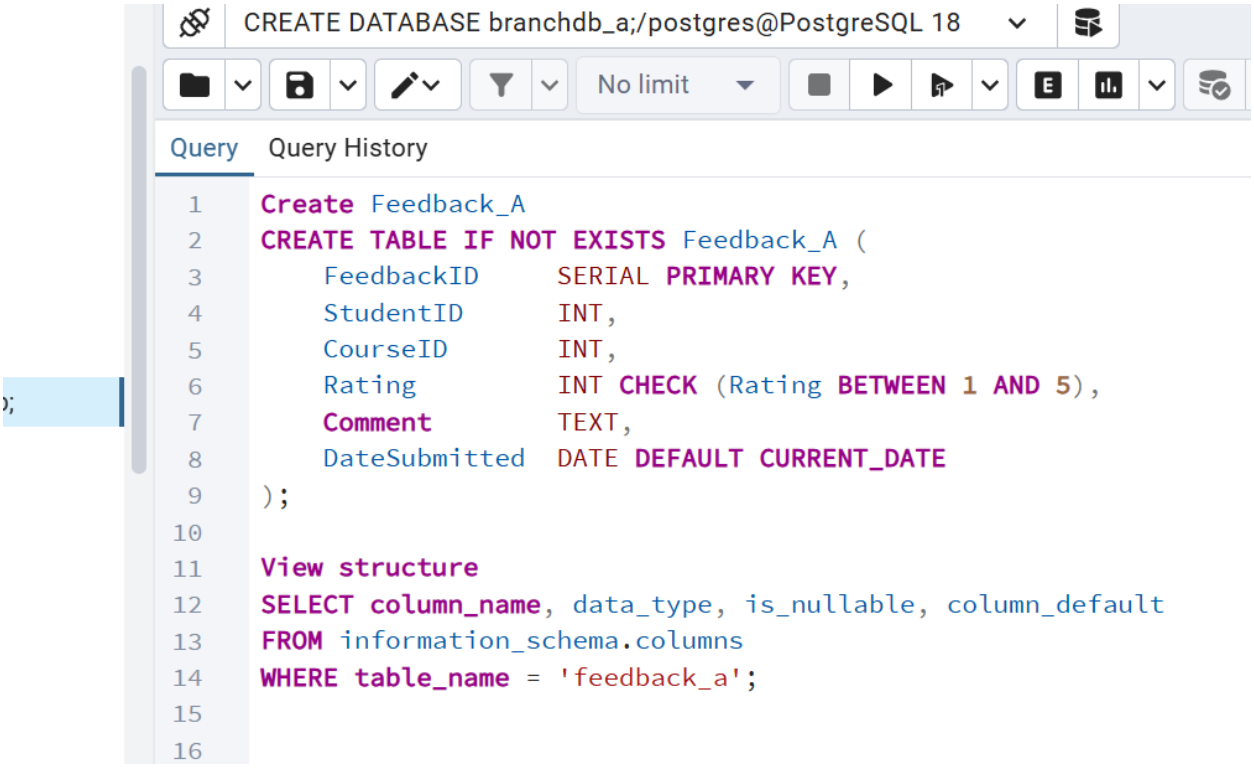
```
INSERT INTO Feedback_B (StudentID, CourseID, Rating, Co
VALUES
(6, 201, 4, 'Good course'),
(7, 202, 5, 'Outstanding'),
(8, 203, 3, 'Average'),
(9, 204, 4, 'Enjoyed it'),
(10, 205, 1, 'Needs improvement');

 Check rows
SELECT * FROM Feedback_B;
```

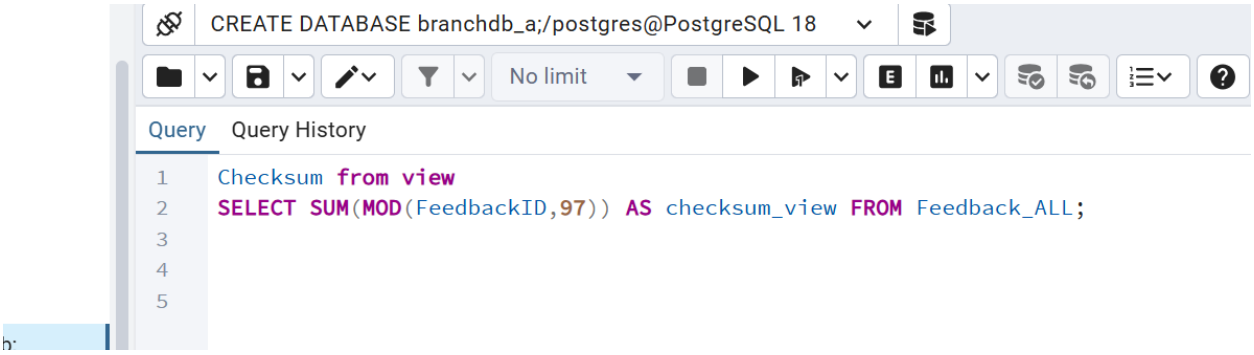
Data Output | Messages | Notifications

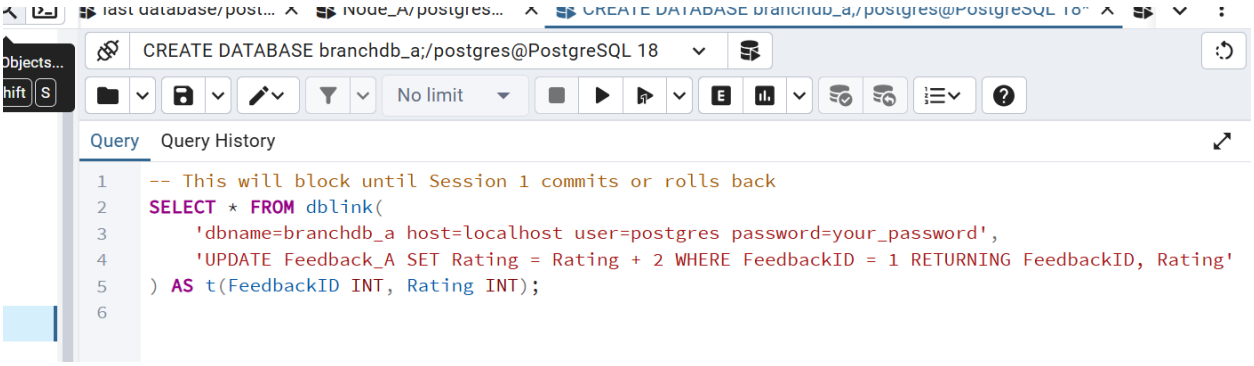
```
NOTICE:  extension "dblink" already exists, skipping
CREATE EXTENSION

Query returned successfully in 35 msec.
```

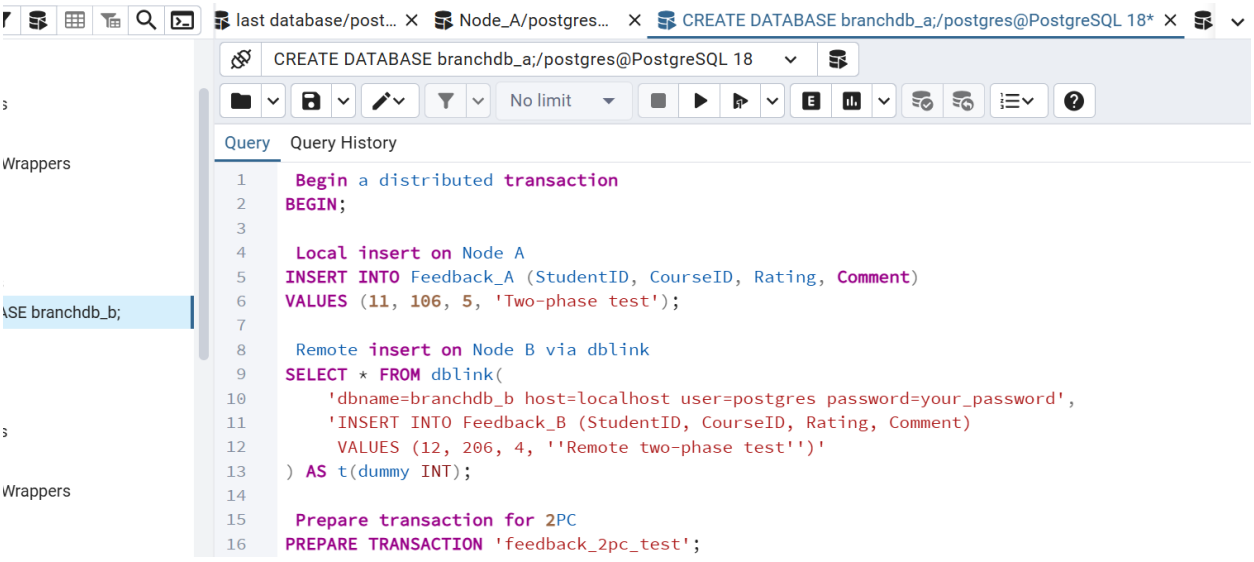
Query | Query History

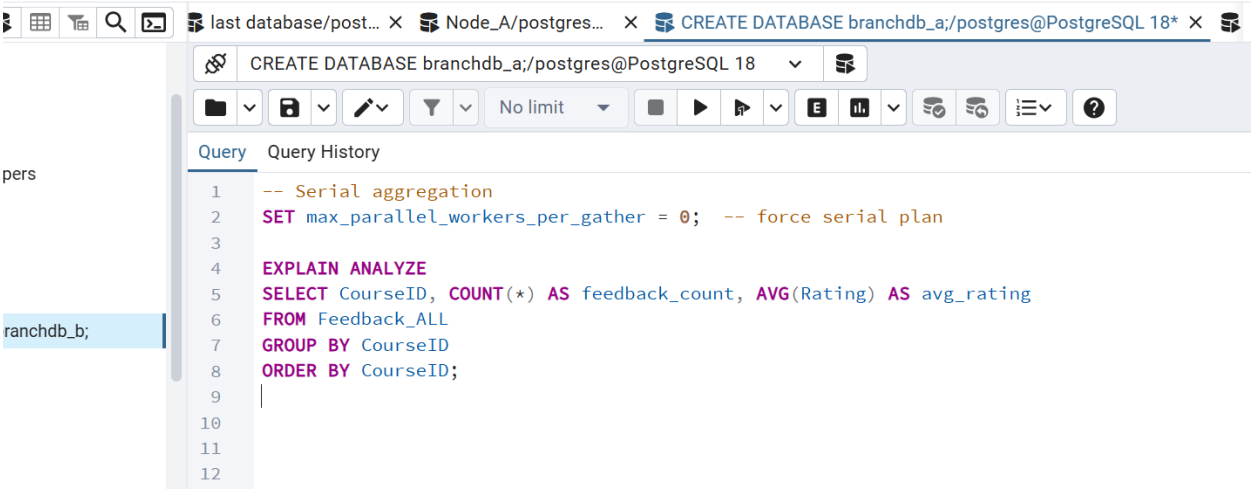
```
INSERT INTO Feedback_A (StudentID, CourseID, Rating, Comment)
VALUES
(1, 101, 5, 'Excellent course'),
(2, 102, 4, 'Good material'),
(3, 103, 3, 'Average pace'),
(4, 104, 5, 'Loved it'),
(5, 105, 2, 'Too difficult');
Check rows

SELECT * FROM Feedback_A;
;
```

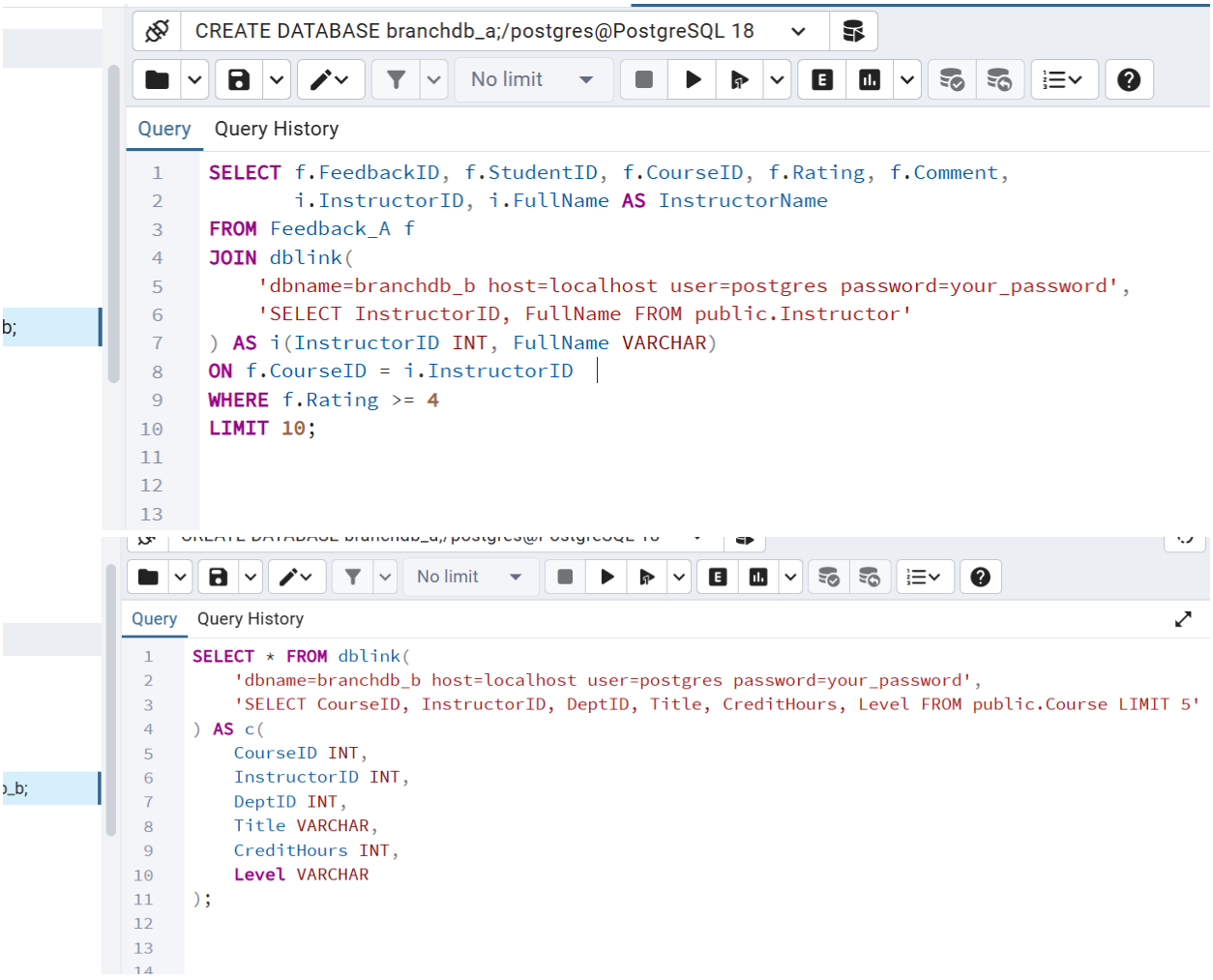

CREATE DATABASE branchdb_a;/postgres@PostgreSQL 18
No limit
Query
Query History
Create Feedback_A
CREATE TABLE IF NOT EXISTS Feedback_A (
FeedbackID SERIAL PRIMARY KEY,
StudentID INT,
CourseID INT,
Rating INT CHECK (Rating BETWEEN 1 AND 5),
Comment TEXT,
DateSubmitted DATE DEFAULT CURRENT_DATE
);
View structure
SELECT column_name, data_type, is_nullable, column_default
FROM information_schema.columns
WHERE table_name = 'feedback_a';

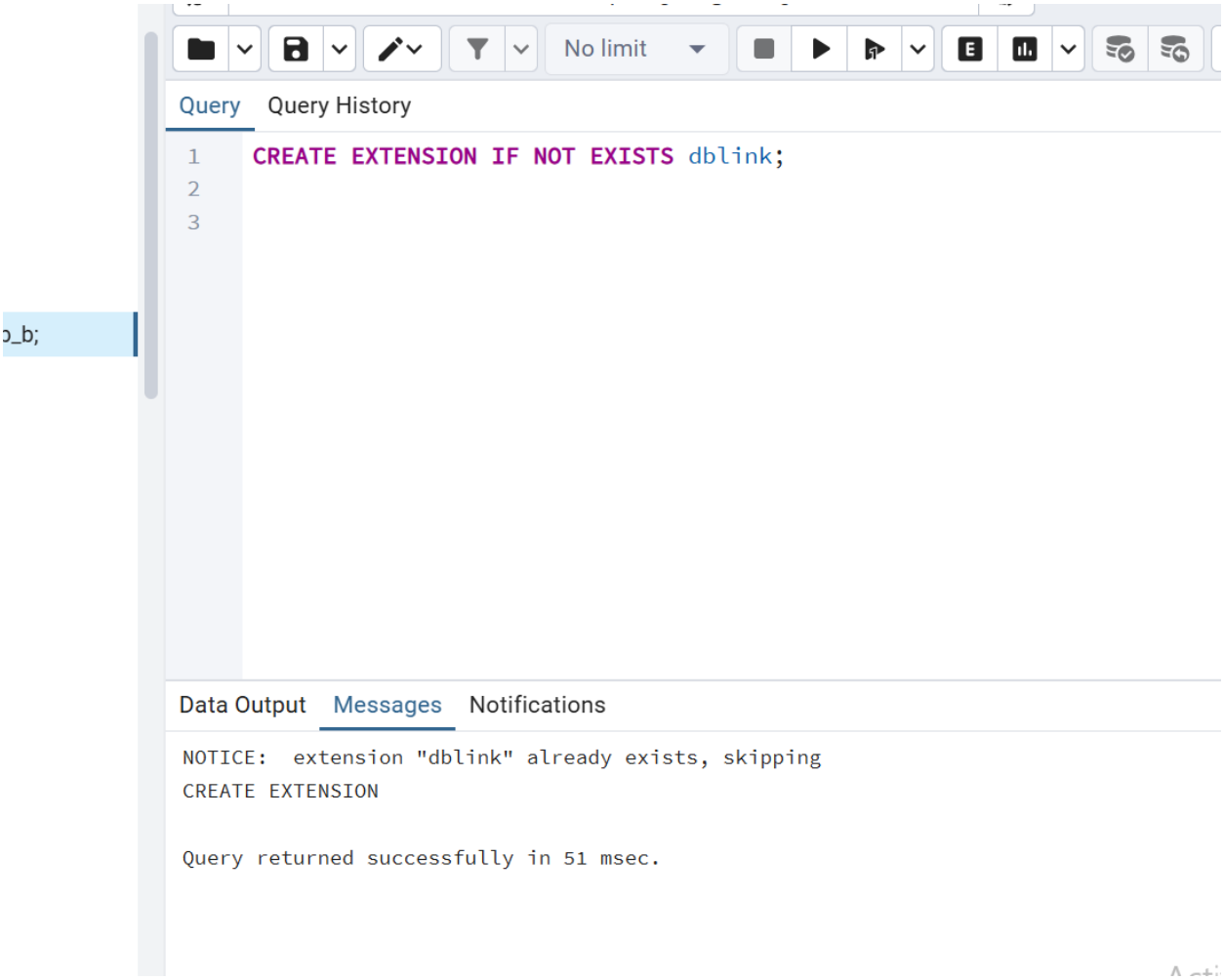

CREATE DATABASE branchdb_a;/postgres@PostgreSQL 18
No limit
Query
Query History
Checksum from view
SELECT SUM(MOD(FeedbackID,97)) AS checksum_view FROM Feedback_ALL;

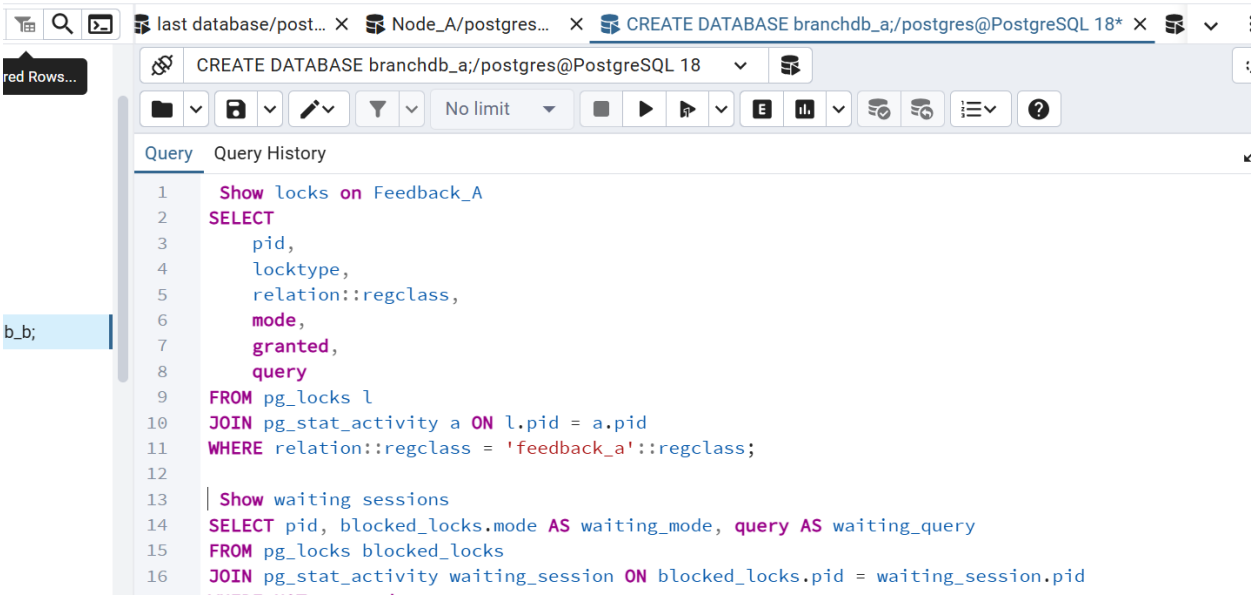

CREATE DATABASE branchdb_a;/postgres@PostgreSQL 18
No limit
Query
Query History
-- This will block until Session 1 commits or rolls back
SELECT * FROM dblink(
'dbname=branchdb_a host=localhost user=postgres password=your_password',
'UPDATE Feedback_A SET Rating = Rating + 2 WHERE FeedbackID = 1 RETURNING FeedbackID, Rating'
) AS t(FeedbackID INT, Rating INT);


last database/post...
Node_A/postgres...
CREATE DATABASE branchdb_a;/postgres@PostgreSQL 18*
CREATE DATABASE branchdb_a;/postgres@PostgreSQL 18
No limit
Query
Query History
Begin a distributed transaction
BEGIN;
Local insert on Node A
INSERT INTO Feedback_A (StudentID, CourseID, Rating, Comment)
VALUES (11, 106, 5, 'Two-phase test');
Remote insert on Node B via dblink
SELECT * FROM dblink(
'dbname=branchdb_b host=localhost user=postgres password=your_password',
'INSERT INTO Feedback_B (StudentID, CourseID, Rating, Comment)
VALUES (12, 206, 4, ''Remote two-phase test'')'
) AS t(dummy INT);
Prepare transaction for 2PC
PREPARE TRANSACTION 'feedback_2pc_test';


last database/post...
Node_A/postgres...
CREATE DATABASE branchdb_a;/postgres@PostgreSQL 18*
CREATE DATABASE branchdb_a;/postgres@PostgreSQL 18
No limit
Query
Query History
-- Serial aggregation
SET max_parallel_workers_per_gather = 0; -- force serial plan
EXPLAIN ANALYZE
SELECT CourseID, COUNT(*) AS feedback_count, AVG(Rating) AS avg_rating
FROM Feedback_ALL
GROUP BY CourseID
ORDER BY CourseID;


CREATE DATABASE branchdb_a;/postgres@PostgreSQL 18
No limit
Query
Query History
SELECT f.FeedbackID, f.StudentID, f.CourseID, f.Rating, f.Comment,
i.InstructorID, i.FullName AS InstructorName
FROM Feedback_A f
JOIN dblink(
'dbname=branchdb_b host=localhost user=postgres password=your_password',
'SELECT InstructorID, FullName FROM public.Instructor'
) AS i(InstructorID INT, FullName VARCHAR)
ON f.CourseID = i.InstructorID
WHERE f.Rating >= 4
LIMIT 10;
No limit
Query
Query History
SELECT * FROM dblink(
'dbname=branchdb_b host=localhost user=postgres password=your_password',
'SELECT CourseID, InstructorID, DeptID, Title, CreditHours, Level FROM public.Course LIMIT 5'
) AS c(
CourseID INT,
InstructorID INT,
DeptID INT,
Title VARCHAR,
CreditHours INT,
Level VARCHAR
);


No limit
Query
Query History
1 CREATE EXTENSION IF NOT EXISTS dblink;
2
3
Data Output
Messages
Notifications
NOTICE: extension "dblink" already exists, skipping
CREATE EXTENSION
Query returned successfully in 51 msec.


last database/post...
Node_A/postgres...
CREATE DATABASE branchdb_a;/postgres@PostgreSQL 18*
CREATE DATABASE branchdb_a;/postgres@PostgreSQL 18
No limit
Query
Query History
1 Show locks on Feedback_A
2 SELECT
3 pid,
4 locktype,
5 relation::regclass,
6 mode,
7 granted,
8 query
9 FROM pg_locks l
10 JOIN pg_stat_activity a ON l.pid = a.pid
11 WHERE relation::regclass = 'feedback_a'::regclass;
12
13 Show waiting sessions
14 SELECT pid, blocked_locks.mode AS waiting_mode, query AS waiting_query
15 FROM pg_locks blocked_locks
16 JOIN pg_stat_activity waiting_session ON blocked_locks.pid = waiting_session.pid