PostgreSQL 빠른 참조

테이블, 쿼리, 조인, 인덱스, JSON, 역할

연결

명령행

```
psql -U postgres
psql -h localhost -p 5432 -U user -d mydb
psql "postgresql://user:pass@host:5432/mydb"
```

psql 메타 명령어

`\l`	데이터베이스 목록
`\c dbname`	데이터베이스 연결
`\dt`	테이블 목록
`\d tablename`	테이블 구조 설명
`\dn`	스키마 목록
`\du`	역할 목록
`\q`	psql 종료
`\i file.sql`	SQL 파일 실행

테이블 및 스키마

테이블 생성

```
CREATE TABLE users (
  id SERIAL PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  email TEXT UNIQUE,
  created_at TIMESTAMPTZ DEFAULT NOW()
);
```

스키마 작업

```
CREATE SCHEMA app;
CREATE TABLE app.users (id SERIAL PRIMARY KEY);
SET search_path TO app, public;
DROP SCHEMA app CASCADE;
```

테이블 변경

```
ALTER TABLE users ADD COLUMN age INT;
ALTER TABLE users ALTER COLUMN name TYPE TEXT;
ALTER TABLE users DROP COLUMN age;
ALTER TABLE users RENAME TO customers;
```

데이터 타입

숫자형

`INTEGER / INT`	4바이트 정수
`BIGINT`	8바이트 정수
`SERIAL`	자동 증가 정수
`NUMERIC(p,s)`	정밀 숫자 (예: NUMERIC(10,2))
`REAL / DOUBLE PRECISION`	부동소수점 (4 / 8바이트)
`BOOLEAN`	true / false / null

문자열 및 바이너리

`TEXT`	가변 길이 무제한 텍스트
`VARCHAR(n)`	최대 n자 가변 텍스트
`CHAR(n)`	고정 길이 텍스트
`BYTEA`	바이너리 데이터
`UUID`	128비트 범용 고유 식별자

날짜, JSON 및 배열

`DATE`	달력 날짜
`TIMESTAMPTZ`	시간대 포함 타임스탬프
`INTERVAL`	시간 간격 (예: '2 days')
`JSONB`	바이너리 JSON (인덱스 가능)
`INT[] / TEXT[]`	배열 타입

쿼리

삽입

```
INSERT INTO users (name, email)
VALUES ('Alice', 'alice@example.com')
RETURNING id;

INSERT INTO users (name, email) VALUES
('Bob', 'bob@ex.com'),
('Carol', 'carol@ex.com');
```

조회

```
SELECT * FROM users WHERE id = 1;
SELECT name, email FROM users
ORDER BY name LIMIT 10 OFFSET 20;
```

업데이트

```
UPDATE users SET email = 'new@ex.com'
WHERE id = 1 RETURNING *;
```

Upsert

```
INSERT INTO users (email, name)
VALUES ('a@ex.com', 'Alice')
ON CONFLICT (email) DO UPDATE
SET name = EXCLUDED.name;
```

삭제

```
DELETE FROM users WHERE id = 1 RETURNING *;
TRUNCATE TABLE users RESTART IDENTITY;
```

조인 및 서브쿼리

조인 유형

`INNER JOIN`	양쪽 테이블 모두 일치하는 행
`LEFT JOIN`	왼쪽 모든 행 + 일치하는 오른쪽
`RIGHT JOIN`	오른쪽 모든 행 + 일치하는 왼쪽
`FULL OUTER JOIN`	양쪽 테이블의 모든 행
`CROSS JOIN`	카르테시안 곱
`LATERAL JOIN`	외부 행을 참조하는 서브쿼리

CTE (공통 테이블 식)

```
WITH active AS (
  SELECT * FROM users WHERE active = true
)
SELECT a.name, o.total
FROM active a
JOIN orders o ON a.id = o.user_id;
```

서브쿼리

```
SELECT name FROM users
WHERE id IN (
  SELECT user_id FROM orders
  WHERE total > 100
);
```

인덱스

생성 및 삭제

```
CREATE INDEX idx_name ON users(name);
CREATE UNIQUE INDEX idx_email ON users(email);
CREATE INDEX idx_gin ON posts USING GIN(tags);
DROP INDEX idx_name;
```

인덱스 유형

`B-tree`	기본값, =, <, >, BETWEEN에 적합
`Hash`	동등 비교만
`GIN`	일반화된 역 인덱스 — 배열, JSONB, 전문 검색
`GiST`	일반화된 검색 — 기하학적, 범위
`BRIN`	블록 범위 — 크고 정렬된 테이블

쿼리 분석

```
EXPLAIN ANALYZE
SELECT * FROM users WHERE name = 'Alice';
```

함수 및 프로시저

SQL 함수

```
CREATE FUNCTION active_count()
RETURNS INTEGER AS $$
  SELECT COUNT(*)::INT FROM users
  WHERE active = true;
$$ LANGUAGE sql;
SELECT active_count();
```

PL/pgSQL 함수

```
CREATE FUNCTION greet(name TEXT)
RETURNS TEXT AS $$
BEGIN
  RETURN 'Hello, ' || name;
END;
$$ LANGUAGE plpgsql;
```

유용한 내장 함수

`NOW() / CURRENT_TIMESTAMP`	시간대 포함 현재 타임스탬프
`AGE(ts1, ts2)`	타임스탬프 간 간격
`COALESCE(a, b)`	첫 번째 비null 값
`NULLIF(a, b)`	a = b이면 NULL
`GENERATE_SERIES(1,10)`	순차 값 행 생성
`STRING_AGG(col, ',')`	구분자로 값 연결

역할 및 권한

역할 관리

```
CREATE ROLE app LOGIN PASSWORD 'secret';
ALTER ROLE app SET search_path TO myapp;
DROP ROLE app;
```

권한 부여

```
GRANT ALL ON DATABASE mydb TO app;
GRANT SELECT, INSERT ON users TO reader;
GRANT USAGE ON SCHEMA public TO app;
REVOKE INSERT ON users FROM reader;
```

행 수준 보안

```
ALTER TABLE users ENABLE ROW LEVEL SECURITY;
CREATE POLICY user_own ON users
FOR ALL USING (id = current_setting('app.uid')::INT);
```

JSON 지원

JSONB 연산자

`->'key'`	키로 JSON 값 가져오기 (JSON으로)
`->>'key'`	키로 JSON 값 가져오기 (텍스트로)
`#>'{a,b}'`	경로로 중첩 값 가져오기
`@>`	포함 (왼쪽이 오른쪽 포함)
`?`	키 존재 여부
`||`	JSONB 값 연결

PostgreSQL 빠른 참조

JSONB 쿼리

```
SELECT data->>'name' FROM profiles
WHERE data @> '{"active": true}';

SELECT * FROM profiles
WHERE data ? 'email';
```

JSONB 함수

```
SELECT jsonb_each(data) FROM profiles;
SELECT jsonb_array_elements('[1,2,3]');
SELECT jsonb_set(data, '{name}', '"Alice"')
FROM profiles WHERE id = 1;
```

일반 패턴

트랜잭션

```
BEGIN;
UPDATE accounts SET balance = balance - 100
WHERE id = 1;
UPDATE accounts SET balance = balance + 100
WHERE id = 2;
COMMIT;  -- or ROLLBACK;
```

윈도우 함수

```
SELECT name, salary,
  RANK() OVER (ORDER BY salary DESC),
  AVG(salary) OVER (PARTITION BY dept)
FROM employees;
```

데이터 복사

```
COPY users TO '/tmp/users.csv'
WITH (FORMAT csv, HEADER);
COPY users FROM '/tmp/users.csv'
WITH (FORMAT csv, HEADER);
```

pg_dump 백업

```
pg_dump -U postgres mydb > backup.sql
pg_dump -Fc mydb > backup.dump
pg_restore -d mydb backup.dump
```