THAM KHẢO NHANH POSTGRESQL

Bảng, truy vấn, joins, indexes, JSON, roles

Kết Nối

Dòng Lệnh

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

Lệnh Meta psql

`\l`	Liệt kê databases
`\c dbname`	Kết nối đến database
`\dt`	Liệt kê bảng
`\d tablename`	Mô tả cấu trúc bảng
`\dn`	Liệt kê schemas
`\du`	Liệt kê roles
`\q`	Thoát psql
`\i file.sql`	Thực thi file SQL

Bảng & Schemas

Tạo Bảng

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

Thao Tác Schema

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

Sửa Đổi Bảng

```
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;
```

Kiểu Dữ Liệu

Số

`INTEGER / INT`	Số nguyên 4 byte
`BIGINT`	Số nguyên 8 byte
`SERIAL`	Số nguyên tự động tăng
`NUMERIC(p,s)`	Số chính xác (vd: NUMERIC(10,2))
`REAL / DOUBLE PRECISION`	Số thực dấu phẩy động (4 / 8 byte)
`BOOLEAN`	true / false / null

Chuỗi & Nhị Phân

`TEXT`	Văn bản không giới hạn độ dài
`VARCHAR(n)`	Văn bản độ dài biến đổi tối đa n ký tự
`CHAR(n)`	Văn bản độ dài cố định
`BYTEA`	Dữ liệu nhị phân
`UUID`	ID duy nhất 128-bit

Ngày, JSON & Mảng

`DATE`	Ngày theo lịch
`TIMESTAMPTZ`	Timestamp có múi giờ
`INTERVAL`	Khoảng thời gian (vd: '2 days')
`JSONB`	JSON nhị phân (có thể đánh index)
`INT[] / TEXT[]`	Kiểu mảng

Truy Vấn

Insert

```
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');
```

Select

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

Update

```
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;
```

Delete

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

Joins & Subqueries

Các Loại Join

`INNER JOIN`	Hàng khớp ở cả hai bảng
`LEFT JOIN`	Tất cả hàng trái + khớp phải
`RIGHT JOIN`	Tất cả hàng phải + khớp trái
`FULL OUTER JOIN`	Tất cả hàng từ cả hai bảng
`CROSS JOIN`	Tích Cartesian
`LATERAL JOIN`	Subquery tham chiếu hàng ngoài

CTE (Common Table Expression)

```
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;
```

Subquery

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

Indexes

Tạo & Xóa

```
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;
```

Loại Index

`B-tree`	Mặc định, tốt cho =, <, >, BETWEEN
`Hash`	Chỉ so sánh bằng
`GIN`	Ngược tổng quát — mảng, JSONB, full-text
`GiST`	Tìm kiếm tổng quát — hình học, range
`BRIN`	Block range — bảng lớn đã sắp xếp

Phân Tích Truy Vấn

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

Hàm & Procedures

SQL Function

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

PL/pgSQL Function

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

Hàm Tích Hợp Hữu Dụng

`NOW() / CURRENT_TIMESTAMP`	Timestamp hiện tại có múi giờ
`AGE(ts1, ts2)`	Khoảng thời gian giữa hai timestamps
`COALESCE(a, b)`	Giá trị không null đầu tiên
`NULLIF(a, b)`	NULL nếu a = b
`GENERATE_SERIES(1,10)`	Tạo các hàng giá trị tuần tự
`STRING_AGG(col, ',')`	Nối các giá trị với dấu phân cách

Roles & Quyền

Quản Lý Role

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

Cấp Quyền

```
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;
```

Row-Level Security

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

Hỗ Trợ JSON

Toán Tử JSONB

`->'key'`	Lấy giá trị JSON theo key (dạng JSON)
`->>'key'`	Lấy giá trị JSON theo key (dạng text)
`#>'{a,b}'`	Lấy giá trị lồng theo path
`@>`	Chứa (trái bao gồm phải)
`?`	Key tồn tại
`||`	Nối hai giá trị JSONB

Truy Vấn JSONB

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

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

Hàm JSONB

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

Mẫu Phổ Biến

Giao Dịch

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

Window Functions

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

Sao Chép Dữ Liệu

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

Sao Lưu pg_dump

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