**JSON Split.**

**Handling nested JSON.**

PostgreSQL Parse JSON operation will take your JSON key-value pairs and convert them into a format that is compatible with PostgreSQL’s table structure. You can parse the JSON objects using PostgreSQL’s JSON operators and functions, and then insert the values into your desired PostgreSQL tables for future use.

CREATE TABLE users (

user\_data JSON

);

INSERT INTO

users (user\_data)

VALUES('{ "id": 1, "name":{"fname":"ravathu","lname":"jahnavi"}, "phone": "989-311-5262"}'),

('{ "id": 2, "name":{"fname":"john","lname":"sanders"}, "phone": "466-731-8534"}'),

('{ "id": 3, "name":{"fname":"rob","lname":"william"}, "phone": "656-432-6184"}'),

('{ "id": 4, "name":{"fname":"bob","lname":"roger"}, "phone": "281-359-3811"}'),

('{ "id": 5, "name":{"fname":"chad","lname":"mane"}, "phone": "541-923-6204"}');

SELECT user\_data ->> 'id' as user\_id, json\_extract\_path\_text(user\_data, 'name', 'fname') as first\_name, json\_extract\_path\_text(user\_data, 'name', 'lname') as last\_name from users;

**Experiment with every function taught till now.**

* SELECT **UPPER**(name) AS name\_in\_uppercase FROM users2;
* SELECT name, **LENGTH**(name) AS no\_of\_chars FROM users2 ORDER BY no\_of\_chars;
* SELECT **INITCAP**(first\_name), INITCAP(last\_name), **CONCAT**(first\_name, ' ' ,last\_name) AS full\_name FROM users3;
* SELECT first\_name, **RPAD**( **SUBSTR**(phone\_no::TEXT,1,2), 5, '\*') AS hidden\_phone\_no FROM users3;
* SELECT COALESCE (NULL, 2 , 1);
* SELECT product\_name, (price - **COALESCE**(discount,0)) AS net\_price FROM items;
* SELECT first\_name, **POSITION**('a' in first\_name) AS position\_a FROM users3;
* SELECT first\_name, **MD5**(last\_name) AS password\_set FROM users3;
* SELECT **REGEXP\_MATCHES**('Learning #PostgreSQL #REGEXP\_MATCHES', '#([A-Za-z0-9\_]+)', 'g') AS hashtags;
* SELECT **REGEXP\_REPLACE**('hello this is jahnavi', '\s', '\_', 'g');
* SELECT **REGEXP\_REPLACE**('John Doe','(.\*) (.\*)','\2, \1');
* SELECT **REGEXP\_SPLIT\_TO\_ARRAY**('hello world', '\s');
* SELECT \* FROM **REGEXP\_SPLIT\_TO\_TABLE**('hello world', '\s');
* SELECT **TRIM**('\*\*\*Jahnavi\*\*\*', '\*');
* SELECT **LTRIM**('\*\*\*Jahnavi\*\*\*', '\*');
* SELECT **RTRIM**('\*\*\*Jahnavi\*\*\*', '\*');
* SELECT **CEIL**(1.01);
* SELECT **FLOOR**(1.99);
* SELECT **LEAST**(2,1,100,-1,8);
* SELECT **GREATEST**(3,10,2,8);

**Given a table in SQL, create 2 tables with odd rows and even rows.**

CREATE TABLE users2 (

id INTEGER,

name TEXT

);

INSERT INTO users2 (id, name)

VALUES (1, 'John'),

(2, 'Jane'),

(3, 'Ross'),

(4, 'Ricky'),

(5, 'Chad'),

(6, 'Nas'),

(7, 'Dunker'),

(8, 'Howard'),

(9, 'Imane'),

(10, 'Bob');

SELECT \* FROM users2 WHERE mod(id,2)=0;

SELECT \* FROM users2 WHERE mod(id,2)=1;

**Convert sql table to csv file and output 2 files with even rows and odd rows.**

COPY users2 TO '/Users/jravathu/Desktop/MANOJ\_ASSIGNEMNTS/16-March/input.csv' DELIMITER ',' CSV HEADER;