PAN NUMBER VALIDATION PROJECT USING POSTGRESQL

Presented By

Somya Agrawal

OBJECTIVE

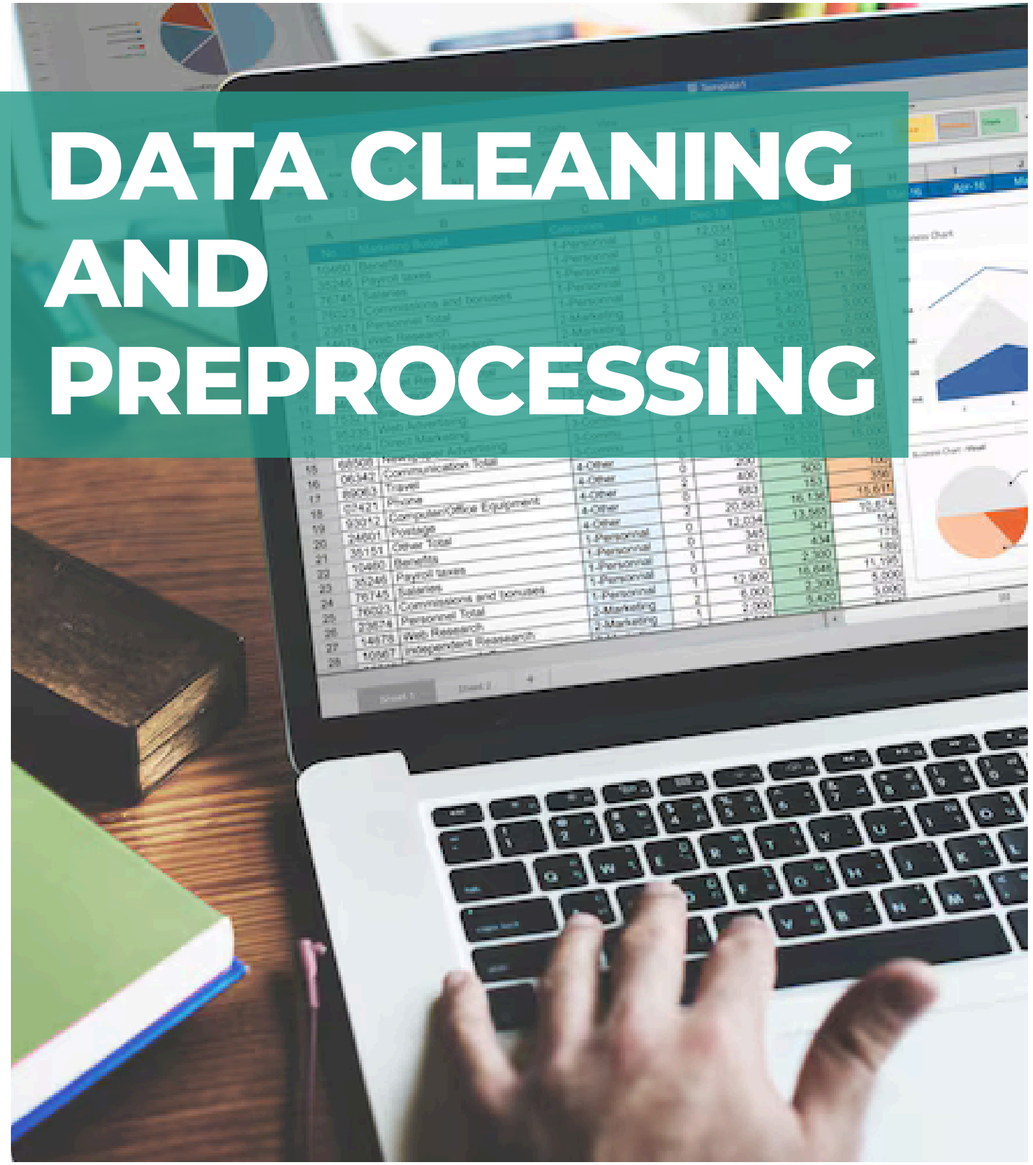
Our task is to clean and validate a dataset containing the Permanent Account Numbers (PAN) of Indian nationals.

The goal is to ensure that each PAN number adheres to the official format and is categorised as either Valid or Invalid.

INSTRUCTIONS

- Data Cleaning and Preprocessing
- PAN Format Validation
- Categorisation : Valid PAN and Invalid PAN
- Tasks : Summary Report

DATA CLEANING AND PREPROCESSING



- **Identify and handle missing data**: PAN numbers may have missing values. These missing values need to be handled appropriately, either by removing rows or imputing values (depending on the context).
- **Check for duplicates:** Ensure there are no duplicate PAN numbers. If duplicates exist, remove them.
- **Handle leading/trailing spaces:** PAN numbers may have extra spaces before or after the actual number. Remove any such spaces.
- **Correct letter case:** Ensure that the PAN numbers are in uppercase letters (if any lowercase letters are present)

SQL QUERIES

```sql
-- 1. Identify and handle missing data
select *
from stg_pan_numbers_dataset
where pan_number is null;
```

Data Output | Messages | Notifications

	pan_number text
1	[null]
2	[null]
3	[null]
4	[null]
5	[null]
6	[null]
7	[null]
8	[null]
9	[null]

Total rows: 965 | Query complete 00:00:00.726

SQL QUERIES

```sql
-- 2. Check for duplicates
select pan_number, count(1)
from stg_pan_numbers_dataset
where pan_number is not null
group by pan_number
having count(1) > 1;
```

Data Output | Messages | Notifications

	pan_number text	count bigint
1	IPSLX475!I	2
2	XTP0675	2
3	JVPYR52307F	4
4	BPWVM28815K	2
5	XVATX221!N	3

Total rows: 5 | Query complete 00:00:00.170

SQL QUERIES

```sql
-- 3. Handle leading/trailing spaces
select *
from stg_pan_numbers_dataset
where pan_number <> trim(pan_number)
```

Data Output | Messages | Notifications

	pan_number text
1	hyuij7902r
2	DOURT5035Y.
3	ZPJQS1155M
4	
5	
6	UQZ4822
7	KEMCQ031!F
8	MGES52860u
9	DLBFA209...

Total rows: 9 | Query complete 00:00:00.158

SQL QUERIES

```sql
-- 4. Correct letter case
select *
from stg_pan_numbers_dataset
where pan_number <> upper(pan_number)
```

Data Output | Messages | Notifications

	pan_number text
1	hvofe5635y
2	hyuij7902r
3	idsmt3429e
4	abkjn5176j
5	vuczu0860t
6	hzvmg3577f
7	owuis0525p
8	HNPC63226j
9	GJRG94249i

Total rows: 990 | Query complete 00:00:00.182

PAN NUMBER FORMAT VALIDATION

- A PAN number is exactly 10 characters long.
- The first five characters should be alphabetic (uppercase letters).
  - Adjacent characters(alphabets) cannot be the same (like AABCD is invalid; AXBCD is valid)
  - All five characters cannot form a sequence (like: ABCDE, BCDEF is invalid; ABCDX is valid)
- The next four characters should be numeric (digits).
  - Adjacent characters(digits) cannot be the same (like 1123 is invalid; 1923 is valid)
  - All four characters cannot form a sequence (like: 1234, 2345)
- The last character should be alphabetic (uppercase letter).

SQL QUERIES

```sql
-- Function to check if adjacent characters are repetitive.
-- Returns true if adjacent characters are adjacent else returns false
create or replace function fn_check_adjacent_repetition(p_str text)
returns boolean
language plpgsql
as $$
begin
    for i in 1 .. (length(p_str) - 1)
    loop
        if substring(p_str, i, 1) = substring(p_str, i+1, 1)
        then
            return true; -- Adjacent characters which are same
        end if;
    end loop;
    return false; -- none of the character adjacent to each other were same
end;
$$
```

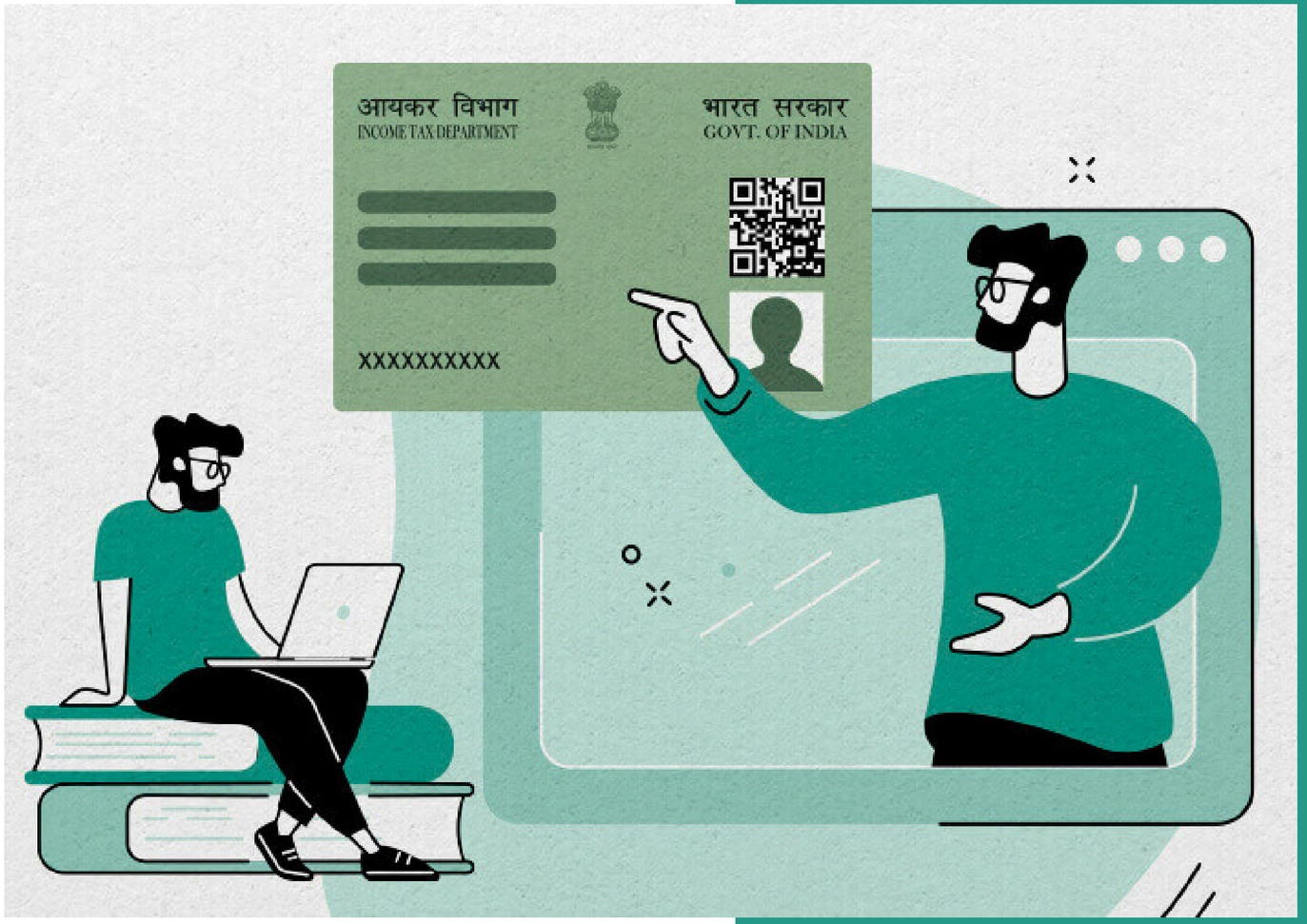
SQL QUERIES

```sql
-- Function to check if characters are sequencial such as ABCDE, LMNOP, XYZ etc.
-- Returns true if characters are sequencial else returns false
CREATE or replace function fn_check_sequence(p_str text)
returns boolean
language plpgsql
as $$
begin
    for i in 1 .. (length(p_str) - 1)
    loop
        if ascii(substring(p_str, i+1, 1)) - ascii(substring(p_str, i, 1)) <> 1
        then
            return false; --string does not form a sequence
        end if;
    end loop;
    return true; --string forms a sequence
end;
$$
```

CATEGORISATION OF PAN NUMBERS:

- **Valid PAN** If the PAN number matches the correct format.

- **Invalid PAN** If the PAN number does not match the correct format, is incomplete, or contains any non-alphanumeric characters.



SQL QUERIES

```sql
-- Valid Invalid PAN categorization
create or replace view vw_valid_invalid_pans
as
with cte_cleaned_pan as
        (select distinct upper(trim(pan_number)) as pan_number
        from stg_pan_numbers_dataset
        where pan_number is not null
        and TRIM(pan_number) <> ''),
    cte_valid_pan as
        (select *
        from cte_cleaned_pan
        where fn_check_adjacent_repetition(pan_number) = 'false'
        and fn_check_sequence(substring(pan_number,1,5)) = 'false'
        and fn_check_sequence(substring(pan_number,6,4)) = 'false'
        and pan_number ~ '^[A-Z]{5}[0-9]{4}[A-Z]$')
select cln.pan_number
, case when vld.pan_number is null
            then 'Invalid PAN'
       else 'Valid PAN'
  end as status
from cte_cleaned_pan cln
left join cte_valid_pan vld on vld.pan_number = cln.pan_number;
```

TASKS

Create a summary report that provides the following:

- Total records processed
- Total valid PANs
- Total invalid PANs
- Total missing or incomplete PANs (if applicable)

SQL QUERY

```sql
-- Summary report
with cte as
    (SELECT
        (SELECT COUNT(*) FROM stg_pan_numbers_dataset) AS total_processed_records,
        COUNT(*) FILTER (WHERE vw.status = 'Valid PAN') AS total_valid_pans,
        COUNT(*) FILTER (WHERE vw.status = 'Invalid PAN') AS total_invalid_pans
    from  vw_valid_invalid_pans vw)
select total_processed_records, total_valid_pans, total_invalid_pans
, total_processed_records - (total_valid_pans+total_invalid_pans) as missing_incomplete_PANS
from cte;
```

Data Output | Messages | Notifications

	total_processed_records bigint	total_valid_pans bigint	total_invalid_pans bigint	missing_incomplete_pans bigint
1	10000	3186	5839	975

Total rows: 1 | Query complete 00:00:00.170

THANK YOU

https://www.linkedin.com/in/somya-agrawal-analyst/

https://neon.zapfolio.in/somya_agrawal936

somya.agrawal936@gmail.com