12
34

Cleaning Data by SQL

Data Befor cleaning

1. **Change Column Names**
2. Check missing values
3. Delete rows if necessary
4. Check misspelling and type
5. Find duplicate values
6. Check consistent fotmat colums
7. Inappropriate null values

Key Benefits

🙏 Thank you for original data set https://www.kaggle.com/datasets/priyamchoksi/credit-card-transactions-dataset

Data Befor cleaning

cards_data.csv

Type of data cleaning check	Example
Dupicates data	client_id = 1,2,2

Type of data cleaning check	Example
Misspeling and types	Viza,creit
Inconsistent format	2023/02,02/2023
Missing values	Blank values
Incorrec data type	1,0,yes,no
Inaccurate data	acc_open_date = 01/01/1900
Inconsistent casting	BangKok,banGkOK
Whitespace issues	“ bangkok”, ” yes”
Unstandardized unit	Bath,Dollar,Yen
Inappropriate null values	They exist in **required fields** (e.g., `id`, `client_id`).

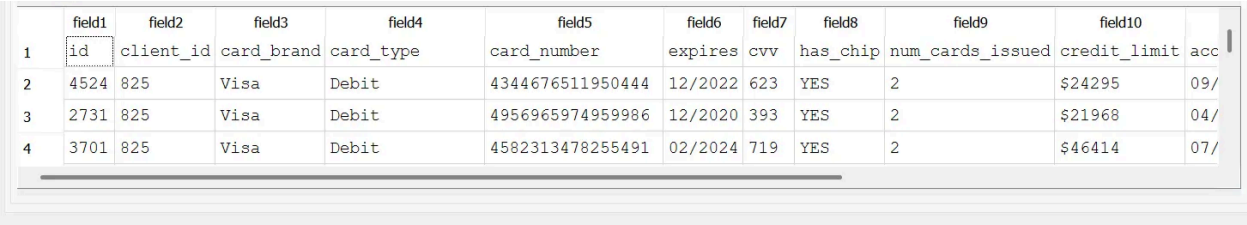
1. Change Column Names

> 💡 In case the column names are not properly formatted or are difficult to use, rename them accordingly.

```
-- Change colums name
--Create new table

CREATE TABLE new_cards_data (
id  INT UNIQUE,
client_id INT,
card_brand TEXT,
card_type TEXT,
card_number TEXT,
expires TEXT,
cvv INT,
has_chip INT,
num_cards_issued INT,
credit_limit REAL,
```

```
acct_open_date TEXT,
year_pin_last_changed TEXT,
card_on_dark_web INT
);
---------------------------------------
--input VALUES in dataset
INSERT INTO new_cards_data (
                        id, client_id,
                        card_brand,
                        card_type,
                        card_number,
                        expires,
                        cvv,has_chip,
                        num_cards_issued,
                        credit_limit,
                        acct_open_date,
                        year_pin_last_changed,
                        card_on_dark_web)
SELECT field1,field2,field3,field4,field5,field6,field7,
       field8,field9,field10,field11,field12,field13
FROM cards_data;
```

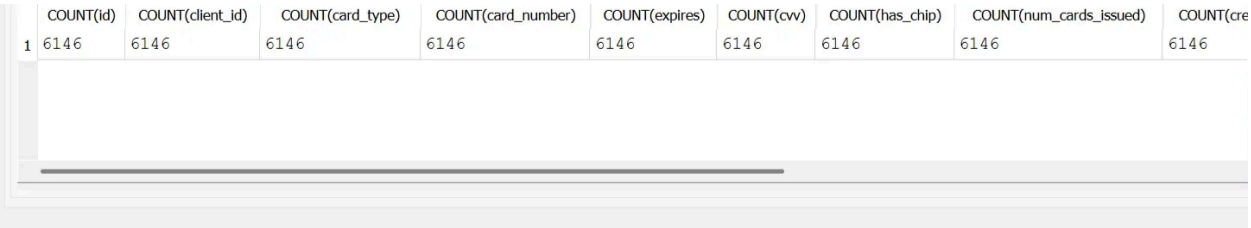


	field1	field2	field3	field4	field5	field6	field7	field8	field9	field10	
1	id	client_id	card_brand	card_type	card_number	expires	cvv	has_chip	num_cards_issued	credit_limit	acc
2	4524	825	Visa	Debit	4344676511950444	12/2022	623	YES	2	$24295	09/
3	2731	825	Visa	Debit	4956965974959986	12/2020	393	YES	2	$21968	04/
4	3701	825	Visa	Debit	4582313478255491	02/2024	719	YES	2	$46414	07/

2. Check missing values

There are many ways to check for missing values. The code below demonstrates one approach by counting the number of missing values in each column.

```
-- count rows in each colums for checking missing values
SELECT
    COUNT(id),
    COUNT(client_id),
    COUNT(card_type),
    COUNT(card_number),
    COUNT(expires),
    COUNT(cvv),
    COUNT(has_chip),
    COUNT(num_cards_issued),
    COUNT(credit_limit),
    COUNT(acct_open_date) ,
    COUNT(year_pin_last_changed),
    COUNT(card_on_dark_web)
FROM new_cards_data;
```



	COUNT(id)	COUNT(client_id)	COUNT(card_type)	COUNT(card_number)	COUNT(expires)	COUNT(cvv)	COUNT(has_chip)	COUNT(num_cards_issued)	COUNT(cre
1	6146	6146	6146	6146	6146	6146	6146	6146	6146

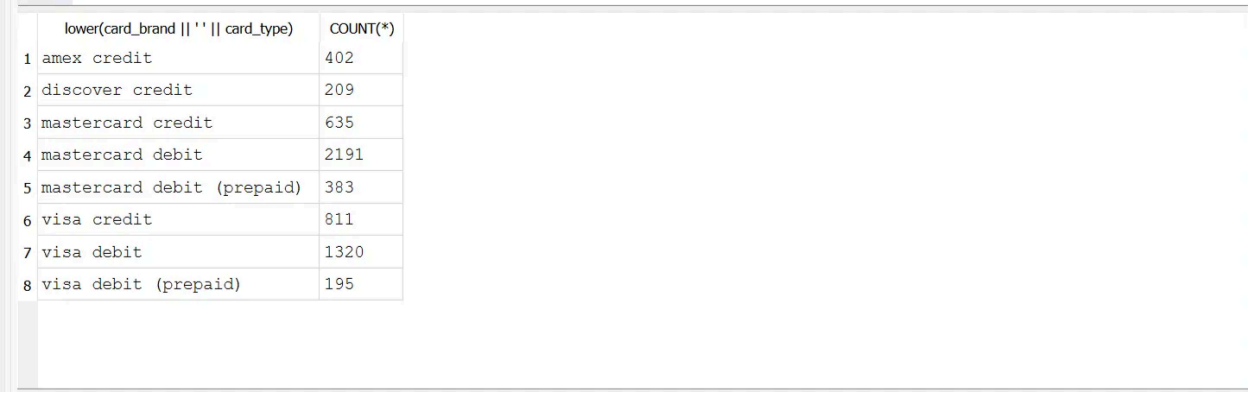
3. Delete rows if necessary

```
DELETE FROM new_cards_data WHERE client_id IS NULL;
```

4. Check misspelling and type

💡 The code below shows how to find distinct values in the columns `card_brand` and `card_type`. If there are variations in spelling, they might indicate misspellings or inconsistencies in the data.

```
--find miss spelling from text colums
SELECT DISTINCT lower(card_brand || ' ' || card_type), COUNT(*)
FROM new_cards_data
GROUP BY lower(card_brand || ' ' || card_type);
Find duplicate data
```



	lower(card_brand || ' ' || card_type)	COUNT(*)
1	amex credit	402
2	discover credit	209
3	mastercard credit	635
4	mastercard debit	2191
5	mastercard debit (prepaid)	383
6	visa credit	811
7	visa debit	1320
8	visa debit (prepaid)	195

5. Find duplicate values

💡 Some columns in the dataset should not have duplicate values, such as `client_id` and `card_number`, as they are expected to be unique.

```
--Find duplicate data
SELECT client_id, card_number, COUNT(*)
FROM new_cards_data
GROUP BY client_id, card_number
HAVING count(*)>1;
```

6. Check consistent fotmat colums

Check for consistent formatting in columns such as `expires` and `acct_open_date`. All values in the same column should follow the same format to ensure data integrity.

```
-- check consistent format date colums
SELECT *
FROM new_cards_data
WHERE expires || '' ||acct_open_date LIKE '__/____'
  AND CAST(SUBSTR(expires, 1, 2) AS INTEGER) > 12;
```

	expires	cvv	has_chip	ı_cards_iss	credit_limit	acct_open_date
4	12/2022	623	YES	2	$24295	09/2002
5	12/2020	393	YES	2	$21968	04/2014
.	02/2024	719	YES	2	$46414	07/2003
7	08/2024	693	NO	1	$12400	01/2003
.	03/2009	75	YES	1	$28	09/2008
3	09/2003	736	YES	1	$27500	09/2003
.	07/2022	972	YES	2	$28508	02/2011
3	06/2022	48	YES	2	$9022	07/2003
5	11/2020	722	YES	2	$54	06/2010
.	02/2023	908	YES	1	$99	07/2006

7. Inappropriate null values

Some columns, such as Yes/No columns, should not contain NULL values. These columns should have only two possible values to maintain data consistency.

```
--check Inappropriate null values
SELECT has_chip, card_on_dark_web
FROM new_cards_data
WHERE LOWER(has_chip) NOT IN ('yes','no');
```

Key Benefits

- **Accuracy**: Clean data ensures reliable decision-making
- **Efficiency**: Clean data enables faster processing
- **Consistency**: Standardized data simplifies analysis

news data →

new_cards_data.csv

🙏 Thank you for original data set https://www.kaggle.com/datasets/priyamchoksi/credit-card-transactions-dataset