FROM SQL TO PANDAS

INDEX

SELECT ALL COLUMNS

SQL

Input table df

CustID	Name
1	Doe
2	Jo
3	Tod

```sql
SELECT *
    FROM df;
```

Output

CustID	Name
1	Doe
2	Jo
3	Tod

pandas

```python
import pandas as pd

data = {
    'CustID': [1, 2, 3],
    'Name': ['Doe', 'Jo', 'Tod']
}


df = pd.DataFrame(data)
```

```python
print(df)
```

output

	CustID	Name
0	1	Doe
1	2	Jo
2	3	Tod

SELECT SINGLE COLUMN

SQL

```sql
CREATE TABLE "df" (
"CustID"  INTEGER,
"Name"    VARCHAR(10)
);

INSERT INTO "df" VALUES
(1, 'Doe'),
(2, 'Jo'),
(3, 'Tod')
;
```

```sql
SELECT "Name"
FROM "df"
```

Name
Doe
Jo
Tod

Pandas

```python
import pandas as pd

data = {
    'CustID': [1, 2, 3],
    'Name': ['Doe', 'Jo', 'Tod']
}

df = pd.DataFrame(data)
```

```python
print(df['Name'])
```

```
0     Doe
1      Jo
2     Tod
Name: Name, dtype: object
```

SELECT MULTIPLE COLUMNS

SQL

```sql
CREATE TABLE "df" (
"CustID" INTEGER,
"FirstName"   VARCHAR,
"LastName" VARCHAR
);

INSERT INTO "df" VALUES
(1, 'Doe', 'Pala'),
(2, 'Jo', 'Noice'),
(3, 'Tod', 'Palle')
;


SELECT
        "CustID",
        "FirstName"
  FROM "df";
```

···	CustID	FirstName
	1	Doe
	2	Jo
	3	Tod

pandas

```python
import pandas as pd

df = pd.DataFrame(
    columns = [
        'CustID',
        'FirstName',
        'LastName'
    ]
)

df['CustID'] = [1, 2, 3]

df['FirstName'] = ['Doe', 'Jo', 'Tod']

df['LastName'] = ['Pala', 'Noice', 'Palle']

print(df)
```

```
  CustID FirstName LastName
0      1       Doe     Pala
1      2        Jo    Noice
2      3       Tod    Palle
```

```python
print(df[['CustID', 'FirstName']])
```

```
  CustID FirstName
0      1       Doe
1      2        Jo
2      3       Tod
```

FILTER ROWS

SQL

```sql
CREATE TABLE "df" (
"CustID"  INTEGER,
"Name"    VARCHAR(10)
);

INSERT INTO "df" VALUES
(1, 'Doe'),
(2, 'Jo'),
(3, 'Tod')
;

SELECT *
  FROM "df"
 WHERE "CustID" = '2';
```

CustID	Name	...
2	Jo	

Pandas

```python
import pandas as pd

df = pd.DataFrame(
    columns = ['CustID', 'Name']
)

df['CustID'] = [1, 2, 3]

df['Name'] = ['Doe', 'Jo', 'Tod']
```

```python
df[df['CustID'] == 2]
```

	CustID	Name
1	2	Jo

TOP N ROWS

SQL

```sql
CREATE TABLE "df" (
"CustID" INTEGER,
"Name"   VARCHAR(10)
);

INSERT INTO "df" VALUES
(1, 'Doe'),
(2, 'Jo'),
(3, 'Tod')
;
```

```sql
SELECT *
    FROM "df"
  LIMIT 1;
```

...	CustID	Name
	1	Doe

pandas

```python
import pandas as pd

df = pd.DataFrame(
    columns = ['CustID', 'Name']
)

df['CustID'] = [1, 2, 3]

df['Name'] = ['Doe', 'Jo', 'Tod']
```

```python
print(df.head(1))
```

```
   CustID Name
0       1  Doe
```

DISTINCT VALUES

SQL

```sql
CREATE TABLE "df" (
"CustID"  INTEGER,
"Name"    VARCHAR
);

INSERT INTO "df" VALUES
(1, 'Doe'),
(2, 'Jo'),
(1, 'Tod')
;


SELECT DISTINCT "CustID"
   FROM "df";
```

CustID
1
2

pandas

```python
import pandas as pd

df = pd.DataFrame(
    columns = ['CustID', 'Name']
)

df['CustID'] = [1, 2, 1]

df['Name'] = ['Doe', 'Jo', 'Tod']

print(df)
```

```
   CustID Name
0       1  Doe
1       2   Jo
2       1  Tod
```

```python
print(df.CustID.unique())
```

```
[1 2]
```

COUNT OF DISTINCT VALUES

SQL

```sql
create table "df" (
"CustID" INTEGER
);

INSERT INTO df values
(10),
(20),
(10);



SELECT COUNT(DISTINCT "CustID")
FROM df;
```

...	COUNT(DISTINCT "CUSTID")
	2

pandas

```python
import pandas as pd

df = pd.DataFrame(
    columns = ['CustID']
)

df['CustID'] = [10, 20, 10]

print(df)
```

```
   CustID
0      10
1      20
2      10
```

```python
print(df.CustID.nunique())
```

```
2
```

COUNT OF TOTAL VALUES

table/dataframe

CustID	Name
10	Doe
20	Jo
30	Tod

```sql
SELECT COUNT(*) * (
    SELECT COUNT(*)
     FROM INFORMATION_SCHEMA.columns
    WHERE TABLE_CATALOG = 'DATABASE_NAME'
      AND TABLE_SCHEMA = 'SCHEMA_NAME'
      AND TABLE_NAME='df'
) AS "Size"
from "df";
```

Size
6

```
df.size
```

6

STRUCTURE OF TABLE

table/dataframe

CustID	Name
10	Doe
20	Jo
30	Tod

```
desc table "df";
```

name	...	type	kind
CustID		NUMBER(38,0)	COLUMN
Name		VARCHAR(20)	COLUMN

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3 entries, 0 to 2
Data columns (total 2 columns):
 #    Column  Non-Null Count   Dtype
---   ------  --------------   -----
 0    CustID  3 non-null       int64
 1    Name    3 non-null       object
dtypes: int64(1), object(1)
memory usage: 176.0+ bytes
```

DESCRIPTIVE STATISTICS

```
In [17]: df

Out[17]: 0    1
         1    2
         2    3
         3    4
         4    5
         Name: AGE, dtype: int64
```

```
In [21]: df.describe()

Out[21]: count    5.0000
         mean     3.0000
         std      1.5811
         min      1.0000
         25%      2.0000
         50%      3.0000
         75%      4.0000
         max      5.0000
         Name: AGE, dtype: float64
```

AGE
1
2
3
4
5

```sql
SELECT

    COUNT(age) AS "count"
,   AVG(age) AS "mean"
,   STDDEV(age) as "std"
,   MIN(age) as "min"
,   PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY age) "25%"
,   PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY age) "50%"
,   PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY age) "75%"
,   MAX(age) as "max"

FROM desc_stats;
```

...	count	mean	std	min	25%	50%	75%	max
	5	3.000000	1.58113883	1	2.000	3.000	4.000	5

GROUP BY SINGLE COLUMN

SQL

```sql
CREATE TABLE "df" (
 "Gender" VARCHAR(1)
,"Population" INTEGER
);

INSERT INTO "df" VALUES
('M', 1),
('F', 1),
('M', 0),
('F', 1)
;


 SELECT
         "Gender"
,        SUM("Population")
  FROM "df"
 GROUP BY "Gender"
  ;
```

Gender	Population
M	1
F	2

pandas

```python
import pandas as pd

df = {
    'Gender': ['M', 'F', 'M', 'F'],
    'Population': [1, 1, 0, 1]
}

df = pd.DataFrame(df)

df
```

	Gender	Population
0	M	1
1	F	1
2	M	0
3	F	1

```python
print(df.groupby('Gender').sum());
```

```
        Population
Gender
F                2
M                1
```

SORT BY SINGLE COLUMN

SQL

```sql
create or replace table "df" (
"ID" INTEGER,
"Name" VARCHAR(10)
);

INSERT INTO "df" values
(5, 'Joe'),
(2, 'Doe'),
(4, 'Paula'),
(3, 'John'),
(1, 'Terry')
;

SELECT *
FROM "df"
ORDER BY "ID";
```

...	ID	Name
	1	Terry
	2	Doe
	3	John
	4	Paula
	5	Joe

pandas

```python
import pandas as pd

df = {
    'ID': [5, 2, 4, 3, 1],
    'NAME': ['Joe', 'Doe', 'Paula', 'John', 'Terry']
}
```

```python
df = pd.DataFrame(df)

df.sort_values(by=['ID'])
```

	ID	NAME
4	1	Terry
1	2	Doe
3	3	John
2	4	Paula
0	5	Joe

SORT BY MULTIPLE COLUMNS

SQL

```sql
create or replace table "df" (
"ID"      INTEGER,
"Name"      VARCHAR(10),
"AGE"      INTEGER
);

INSERT INTO "df" values
(5, 'Joe', 20),
(2, 'Doe', 50),
(2, 'Paula', 10),
(1, 'John', 40),
(1, 'Terry', 30)
;
```

```sql
SELECT *
FROM "df"
ORDER BY "ID", "AGE";
```

...	ID	Name	AGE
	1	Terry	30
	1	John	40
	2	Paula	10
	2	Doe	50
	5	Joe	20

pandas

```python
import pandas as pd

df = {
    'ID': [5, 2, 2, 1, 1],
    'NAME': ['Joe', 'Doe', 'Paula', 'John', 'Terry'],
    'AGE': [20, 50, 10, 40, 30]
}
```

```python
df = pd.DataFrame(df)

df.sort_values(by=['ID', 'AGE'])
```

	ID	NAME	AGE
4	1	Terry	30
3	1	John	40
2	2	Paula	10
1	2	Doe	50
0	5	Joe	20

Sorted by default in ascending order

COUNT OF UNIQUE VALUES

SQL

```sql
create or replace table "df" (
"NAME"    VARCHAR(10)
);

INSERT INTO "df" values
('Joe'),
('Doe'),
('Paula'),
('Joe'),
('Doe')
;


SELECT "NAME", COUNT(*)
FROM "df"
GROUP BY "NAME"
ORDER BY COUNT(*) DESC;
```

NAME	···	COUNT(*)
Joe		2
Doe		2
Paula		1

Pandas

```python
import pandas as pd

df = ['Joe', 'Doe', 'Paula', 'Joe', 'Doe']

df = pd.DataFrame(df)
```

```python
df.value_counts()
```

```
Doe        2
Joe        2
Paula      1
Name: count, dtype: int64
```

DROP ROW - ALL COLUMNS DUPLICATED

SQL

```sql
create or replace table "df" (
"ID"       INTEGER,
"NAME"     VARCHAR(10)
);

INSERT INTO "df" values
(1, 'Joe'),
(2, 'Jack'),
(3, 'Paula'),
(1, 'Joe')
;

SELECT DISTINCT *
FROM "df"
;
```

...	ID	NAME
	1	Joe
	2	Jack
	3	Paula

pandas

```python
import pandas as pd

df = {
    'ID': [1, 2, 3, 1],
    'Name': ['Joe', 'Jack', 'Paul', 'Joe']
}

df = pd.DataFrame(df)

df
```

	ID	Name
0	1	Joe
1	2	Jack
2	3	Paul
3	1	Joe

```python
df.drop_duplicates()
```

	ID	Name
0	1	Joe
1	2	Jack
2	3	Paul

DROP ROW - KEY COLUMN DUPLICATED

SQL

```sql
create or replace table "df" (
"ID"        INTEGER,
"NAME"      VARCHAR(10)
);

INSERT INTO "df" values
(1, 'Joe'),
(2, 'Jack'),
(3, 'Paula'),
(1, 'Doe')
;


DELETE FROM "df" T1
USING
(
    SELECT
        ID,
        NAME
    FROM "df"
    QUALIFY ROW_NUMBER() OVER (PARTITION BY ID ORDER BY ID ASC) = '2'
) T2
WHERE T1."ID" = T2."ID" AND T1."NAME" = T2."NAME"
;

SELECT *
FROM "df"
;
```

...	ID	NAME
	1	Joe
	2	Jack
	3	Paula

pandas

```python
import pandas as pd
```

```python
df = {
    'ID': [1, 2, 3, 1],
    'Name': ['Joe', 'Jack', 'Paul', 'Doe']
}
```

```python
df = pd.DataFrame(df)
```

```python
df
```

	ID	Name
0	1	Joe
1	2	Jack
2	3	Paul
3	1	Doe

```python
df.drop_duplicates(subset=['ID'])
```

	ID	Name
0	1	Joe
1	2	Jack
2	3	Paul

INNER JOIN

SQL

```sql
create or replace table "df1" (
"ID"        INTEGER,
"NAME"      VARCHAR(10)
);
INSERT INTO "df1" values
(1, 'Joe'),
(2, 'Jack'),
(3, 'Paula');
create or replace table "df2" (
"ID"        INTEGER,
"AGE"       INTEGER
);
INSERT INTO "df2" values
(1, 10),
(2, 20),
(4, 40);
SELECT "df1".ID, NAME, AGE
FROM "df1"
INNER JOIN "df2"
ON "df1".ID = "df2".ID;
```

...	ID	NAME	AGE
	1	Joe	10
	2	Jack	20

pandas

```python
import pandas as pd
```

```python
df1 = {
    'ID': [1, 2, 3],
    'Name': ['Joe', 'Jack', 'Paula']
}

df1 = pd.DataFrame(df1)

df2 = {
    'ID': [1, 2, 4],
    'Age': [10, 20, 40],
}

df2 = pd.DataFrame(df2)
```

```python
df = pd.merge(df1, df2, on='ID', how='inner')

df
```

	ID	Name	Age
0	1	Joe	10
1	2	Jack	20

Uzwal Goud Vaddeboina

LEFT JOIN

SQL

```sql
create or replace table "df1" (
"ID"        INTEGER,
"NAME"      VARCHAR(10)
);
INSERT INTO "df1" values
(1, 'Joe'),
(2, 'Jack'),
(3, 'Paula');
create or replace table "df2" (
"ID"      INTEGER,
"AGE"     INTEGER
);
INSERT INTO "df2" values
(1, 10),
(2, 20),
(4, 40);
SELECT "df1".ID, NAME, AGE
FROM "df1"
LEFT JOIN "df2"
ON "df1".ID = "df2".ID;
```

ID	NAME	AGE
1	Joe	10
2	Jack	20
3	Paula	null

pandas

```python
import pandas as pd
```

```python
df1 = {
    'ID': [1, 2, 3],
    'Name': ['Joe', 'Jack', 'Paula']
}

df1 = pd.DataFrame(df1)

df2 = {
    'ID': [1, 2, 4],
    'Age': [10, 20, 40],
}

df2 = pd.DataFrame(df2)
```

```python
df = pd.merge(df1, df2, on='ID', how='left')

df
```

	ID	Name	Age
0	1	Joe	10.0
1	2	Jack	20.0
2	3	Paula	NaN

RIGHT JOIN

SQL

```sql
create or replace table "df1" (
"ID"      INTEGER,
"NAME"    VARCHAR(10)
);
INSERT INTO "df1" values
(1, 'Joe'),
(2, 'Jack'),
(3, 'Paula');
create or replace table "df2" (
"ID"      INTEGER,
"AGE"     INTEGER
);
INSERT INTO "df2" values
(1, 10),
(2, 20),
(4, 40);
SELECT "df2".ID, NAME, AGE
FROM "df1"
RIGHT JOIN "df2"
ON "df1".ID = "df2".ID;
```

ID	NAME	...	AGE
1	Joe		10
2	Jack		20
4	null		40

pandas

```python
import pandas as pd
```

```python
df1 = {
    'ID': [1, 2, 3],
    'Name': ['Joe', 'Jack', 'Paula']
}

df1 = pd.DataFrame(df1)

df2 = {
    'ID': [1, 2, 4],
    'Age': [10, 20, 40],
}

df2 = pd.DataFrame(df2)
```

```python
df = pd.merge(df1, df2, on='ID', how='right')

df
```

	ID	Name	Age
0	1	Joe	10
1	2	Jack	20
2	4	NaN	40

Uzwal Goud Vaddeboina

FULL JOIN

SQL

```sql
create or replace table "df1" (
"ID"      INTEGER,
"NAME"    VARCHAR(10)
);
INSERT INTO "df1" values
(1, 'Joe'),
(2, 'Jack'),
(3, 'Paula');
create or replace table "df2" (
"ID"      INTEGER,
"AGE"     INTEGER
);
INSERT INTO "df2" values
(1, 10),
(2, 20),
(4, 40);
SELECT COALESCE("df1".ID, "df2".ID) AS ID,
        NAME,
        AGE
FROM "df1"
FULL JOIN "df2"
ON "df1".ID = "df2".ID;
ORDER BY 1;
```

ID	NAME	AGE
1	Joe	10
2	Jack	20
3	Paula	null
4	null	40

pandas

```python
import pandas as pd
```

```python
df1 = {
    'ID': [1, 2, 3],
    'Name': ['Joe', 'Jack', 'Paula']
}

df1 = pd.DataFrame(df1)

df2 = {
    'ID': [1, 2, 4],
    'Age': [10, 20, 40],
}

df2 = pd.DataFrame(df2)
```

```python
df = pd.merge(df1, df2, on='ID', how='outer')

df
```

	ID	Name	Age
0	1	Joe	10.0
1	2	Jack	20.0
2	3	Paula	NaN
3	4	NaN	40.0

in Uzwal Goud Vaddeboina

CROSS JOIN

SQL

```sql
create or replace table "df1" (
"ID"       INTEGER,
"NAME"     VARCHAR(10)
);
INSERT INTO "df1" values
(1, 'Joe'),
(2, 'Jack');
create or replace table "df2" (
"ID"       INTEGER,
"AGE"      INTEGER
);
INSERT INTO "df2" values
(1, 10),
(2, 20);
SELECT *
FROM "df1"
CROSS JOIN "df2"
;
```

...	ID	NAME	ID_2	AGE
	1	Joe	1	10
	1	Joe	2	20
	2	Jack	1	10
	2	Jack	2	20

pandas

```python
import pandas as pd
```

```python
df1 = {
    'ID': [1, 2],
    'Name': ['Joe', 'Jack']
}

df1 = pd.DataFrame(df1)

df2 = {
    'ID': ['1', '2'],
    'AGE': [10, 20]
}

df2 = pd.DataFrame(df2)
```

```python
df = pd.merge(df1, df2, how='cross')

df
```

	ID_x	Name	ID_y	AGE
0	1	Joe	1	10
1	1	Joe	2	20
2	2	Jack	1	10
3	2	Jack	2	20

UNION ALL BY SINGLE COLUMN

SQL

```sql
CREATE OR REPLACE TABLE "df1" (
"ID"        INTEGER,
"NAME"      VARCHAR(10)
);
INSERT INTO "df1" values
(1, 'Joe'),
(2, 'Jack');
--

CREATE OR REPLACE TABLE "df2" (
"ID"        INTEGER,
"NAME"      VARCHAR(10)
);
INSERT INTO "df2" values
(1, 'Joe'),
(4, 'Doe');
--

SELECT NAME
FROM "df1"
UNION ALL
SELECT NAME
FROM "df2"
;
```

NAME
Joe
Jack
Joe
Doe

pandas

```python
import pandas as pd
```

```python
df1 = {
    'ID': [1, 2],
    'Name': ['Joe', 'Jack']
}

df1 = pd.DataFrame(df1)

df2 = {
    'ID': [1, 4],
    'Name': ['Joe', 'Doe']
}

df2 = pd.DataFrame(df2)
```

```python
df = pd.concat([df1['Name'], df2['Name']], ignore_index=True)

# ignore_index=True will reindex the dataframe

df
```

```
0       Joe
1       Jack
2       Joe
3       Doe
```

UNION ALL BY ALL COLUMNS

SQL

```sql
CREATE OR REPLACE TABLE "df1" (
"ID"      INTEGER,
"NAME"    VARCHAR(10)
);
INSERT INTO "df1" values
(1, 'Joe'),
(2, 'Jack');
--

CREATE OR REPLACE TABLE "df2" (
"ID"      INTEGER,
"NAME"    VARCHAR(10)
);
INSERT INTO "df2" values
(1, 'Joe'),
(4, 'Doe');
--

SELECT *
FROM "df1"
UNION ALL
SELECT *
FROM "df2"
;
```

ID	NAME
1	Joe
2	Jack
1	Joe
4	Doe

pandas

```python
import pandas as pd
```

```python
df1 = {
    'ID': [1, 2],
    'Name': ['Joe', 'Jack']
}

df1 = pd.DataFrame(df1)

df2 = {
    'ID': [1, 4],
    'Name': ['Joe', 'Doe']
}

df2 = pd.DataFrame(df2)
```

```python
df = pd.concat([df1, df2], ignore_index=True)

# ignore_index=True will reindex the dataframe

df
```

	ID	Name
0	1	Joe
1	2	Jack
2	1	Joe
3	4	Doe

Uzwal Goud Vaddeboina

UNION BY SINGLE COLUMN

SQL

```sql
CREATE OR REPLACE TABLE "df1" (
"ID"      INTEGER,
"NAME"    VARCHAR(10)
);
INSERT INTO "df1" values
(1, 'Joe'),
(2, 'Jack');
--

CREATE OR REPLACE TABLE "df2" (
"ID"      INTEGER,
"NAME"    VARCHAR(10)
);
INSERT INTO "df2" values
(1, 'Joe'),
(4, 'Doe');
--

SELECT NAME
FROM "df1"
UNION
SELECT NAME
FROM "df2"
;
```

NAME ⋯
Joe
Jack
Doe

pandas

```python
import pandas as pd
```

```python
df1 = {
    'ID': [1, 2],
    'Name': ['Joe', 'Jack']
}

df1 = pd.DataFrame(df1)

df2 = {
    'ID': [1, 4],
    'Name': ['Joe', 'Doe']
}

df2 = pd.DataFrame(df2)
```

```python
df = pd.concat([df1['Name'], df2['Name']], ignore_index=True).drop_duplicates()

# ignore_index=True will reindex the dataframe

df
```

```
0      Joe
1     Jack
3      Doe
```

UNION BY ALL COLUMNS

SQL

```sql
CREATE OR REPLACE TABLE "df1" (
"ID"        INTEGER,
"NAME"      VARCHAR(10)
);
INSERT INTO "df1" values
(1, 'Joe'),
(2, 'Jack');
--
CREATE OR REPLACE TABLE "df2" (
"ID"        INTEGER,
"NAME"      VARCHAR(10)
);
INSERT INTO "df2" values
(1, 'Joe'),
(4, 'Doe');
--
SELECT *
FROM "df1"
UNION
SELECT *
FROM "df2"
;
```

ID	NAME
1	Joe
2	Jack
4	Doe

pandas

```python
import pandas as pd
```

```python
df1 = {
    'ID': [1, 2],
    'Name': ['Joe', 'Jack']
}

df1 = pd.DataFrame(df1)

df2 = {
    'ID': [1, 4],
    'Name': ['Joe', 'Doe']
}

df2 = pd.DataFrame(df2)
```

```python
df = pd.concat([df1, df2],ignore_index=True).drop_duplicates()

df
```

	ID	Name
0	1	Joe
1	2	Jack
3	4	Doe

ADD SINGLE COLUMN

SQL

```sql
CREATE TABLE "data" (
    "ID"  INTEGER
,   "NAME" VARCHAR(10)
);

INSERT INTO "data" VALUES
(1, 'Joe'),
(2, 'Jack'),
(3, 'Paula')
;

ALTER TABLE "data" ADD COLUMN AGE INTEGER;

UPDATE "data" SET AGE = 10 WHERE ID = '1';
UPDATE "data" SET AGE = 20 WHERE ID = '2';
UPDATE "data" SET AGE = 40 WHERE ID = '3';

SELECT * FROM "data";
```

...	ID	NAME	AGE
	1	Joe	10
	2	Jack	20
	3	Paula	40

pandas

```python
data = {
    'ID': [1, 2, 3],
    'Name': ['Joe', 'Jack', 'Paula']
}
```

```python
df = pd.DataFrame(data)
```

```python
df
```

	ID	Name
0	1	Joe
1	2	Jack
2	3	Paula

```python
df['Age'] = [10, 20, 40]
```

```python
df
```

	ID	Name	Age
0	1	Joe	10
1	2	Jack	20
2	3	Paula	40

DROP SINGLE COLUMN

SQL

```sql
CREATE or replace TABLE "data" (
     "ID"  INTEGER
,     "NAME" VARCHAR(10)
,     "AGE"  INTEGER
);

INSERT INTO "data" VALUES
(1, 'Joe', '10'),
(2, 'Jack', '20'),
(3, 'Paula', '40')
;

ALTER TABLE "data" DROP COLUMN "AGE";

SELECT * FROM "data";
```

...	ID	NAME
	1	Joe
	2	Jack
	3	Paula

pandas

```python
data = {
    'ID': [1, 2, 3],
    'Name': ['Joe', 'Jack', 'Paula'],
    'Age': [10, 20, 40]
}
```

```python
df = pd.DataFrame(data)
```

```python
df
```

	ID	Name	Age
0	1	Joe	10
1	2	Jack	20
2	3	Paula	40

```python
df.drop(['Age'], axis=1, inplace=True)
```

```python
df
```

	ID	Name
0	1	Joe
1	2	Jack
2	3	Paula

DROP MULTIPLE COLUMNS

SQL

```sql
CREATE or replace TABLE "data" (
      "ID"   INTEGER
,     "NAME" VARCHAR(10)
,     "AGE"  INTEGER
);

INSERT INTO "data" VALUES
(1, 'Joe', '10'),
(2, 'Jack', '20'),
(3, 'Paula', '40')
;

ALTER TABLE "data"
DROP COLUMN "AGE", "NAME";

SELECT * FROM "data";
```

ID
1
2
3

pandas

```python
import pandas as pd
```

```python
data = {
    'ID': [1, 2, 3],
    'Name': ['Joe', 'Jack', 'Paula'],
    'Age': [10, 20, 40]
}
```

```python
df = pd.DataFrame(data)
```

```python
df
```

	ID	Name	Age
0	1	Joe	10
1	2	Jack	20
2	3	Paula	40

```python
df.drop(['Age', 'Name'], axis=1, inplace=True)
```

```python
df
```

	ID
0	1
1	2
2	3

Uzwal Goud Vaddeboina

RENAME SINGLE COLUMN

SQL

```sql
CREATE TABLE "data" (
    "ID"  INTEGER
,   "NAME"  VARCHAR(10)
,   "AGE"  INTEGER
,   "HEIHT"  VARCHAR(10)
);

ALTER TABLE "data"
RENAME COLUMN "HEIHT" to "HEIGHT";

DESC TABLE "data";
```

name	type
ID	NUMBER(38,0)
NAME	VARCHAR(10)
AGE	NUMBER(38,0)
HEIGHT	VARCHAR(10)

Pandas

```python
import pandas as pd
```

```python
data = {
    'ID': [1, 2, 3],
    'Name': ['Joe', 'Jack', 'Paula'],
    'Age': [10, 20, 40],
    'Heiht': ['1.65', '1.78', '1.82']
}
```

```python
df = pd.DataFrame(data)
```

```python
df.rename(columns = {'Heiht':'Height'}, inplace = True)
```

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3 entries, 0 to 2
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   ID      3 non-null      int64
 1   Name    3 non-null      object
 2   Age     3 non-null      int64
 3   Height  3 non-null      object
dtypes: int64(2), object(2)
memory usage: 228.0+ bytes
```

in Uzwal Goud Vaddeboina

RENAME MULTIPLE COLUMNS


```sql
CREATE TABLE "data" (
      "ID" INTEGER
,     "NME" VARCHAR(10)
,     "AGE" INTEGER
,     "HEIHT" VARCHAR(10)
);

ALTER TABLE "data"
RENAME COLUMN "NME" to "NAME";

ALTER TABLE "data"
RENAME COLUMN "HEIHT" to "HEIGHT";

DESC TABLE "data";
```

name	type
ID	NUMBER(38,0)
NAME	VARCHAR(10)
AGE	NUMBER(38,0)
HEIGHT	VARCHAR(10)

```python
import pandas as pd
```

```python
data = {
    'ID': [1, 2, 3],
    'Nme': ['Joe', 'Jack', 'Paula'],
    'Age': [10, 20, 40],
    'Heiht': ['1.65', '1.78', '1.82']
}
```

```python
df = pd.DataFrame(data)
```

```python
df.rename(columns = {'Nme':'Name',
                     'Heiht':'Height'},
          inplace = True)
```

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3 entries, 0 to 2
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   ID      3 non-null      int64
 1   Name    3 non-null      object
 2   Age     3 non-null      int64
 3   Height  3 non-null      object
dtypes: int64(2), object(2)
memory usage: 228.0+ bytes
```

That's a wrap!

If you liked this content, follow **Uzwal** on LinkedIn and click the bell icon for updates.