SQL DATA STANDARDIZATION DOCUMENTATION

DNA LEARNING (WEEKEND - 1)

STRING FUNCTIONS

1. TRIM() - removes trailing and leading spaces from the string
2. LTRIM() - removes trailing or leading space from the beginning of the string
3. RTRIM() - removes trailing or leading spaces from the end of the string
4. UPPER() - Capitalizes the string
5. LOWER() - Makes the string characters into small letters
6. LENGTH() - finds length of string
7. SUBSTRING() - Extracts the part of the string

   SUBSTRING('john', 1, 3) - (string, start position, no of characters to be extracted)

   Ans: joh

8. INSTR() - Gives the position of character in the string.

   INSTR(manhatoo.mackry@gmail.com, '@')

   Ans : 16

9. CHAR_LENGTH() - Finds no of characters in the string
10. CONCAT() - Adds two strings

11. REGEXP_REPLACE() - Removes unwanted characters

    REGEXP_REPLACE(728973jh89u23x534, '^[0-9]', '')

    Ans: 7289738923534

12. LPAD - Adds any character from beginning of string to make it to a specific length

    LPAD(555, 6, 0) - Given number is 555, and result length should be 6 padded up With 0

    Ans : 000555

13. RPAD - Adds any character from the end of string

    RPAD(555, 6, 9) - Given number is 555, and result length should be 6 padded up

S DHAANESH

With 9

Ans : 555999

<u>SQL DATA ANALYSIS (CUSTOMER DATASET)</u>

Dataset - https://github.com/Dhaanesh26/data_standardisation/blob/main/datasets/customer.csv

Q1. Remove extra spaces and fix casing in names

```
SELECT TRIM(UPPER(CustomerName))
```

Q2. Standardize phone number formats

```
SELECT PhoneNumber,
CONCAT(‘+1’,
RIGHT(REGEXP_REPLACE(SUBSTRING_INDEX(PhoneNumber,‘x’,1),‘[^0-9]’, ‘’)), 10)
```

Q3. Check email if @ is present

```
SELECT Email,
        CASE
                WHEN  INSTR(Email, ‘@’) > 0 THEN ‘Valid’
                ELSE ‘Inavalid’
        END AS email_status
FROM customer
```

Q4. Query the domain name from email

```
SELECT Email, SUBSTRING(Email, INSTR(Email, ‘@’) + 1) AS domain_name
FROM customer
```

Q5. Convert string to proper date format

```
SELECT STR_TO_DATE(CreatedDate, ‘%Y-%m-%d’) AS standardised_signup_date
FROM customer
```

Q6. Standardize categorical data using CASE statements

```
SELECT isActive,
        CASE
                WHEN UPPER(isActive) IN (‘TRUE’, ‘T’) THEN ‘True’
                ELSE ‘False’
```

END AS customer_status
FROM customer;

Q7. Consistent granularity for analysis (Extra)

SELECT DATE_FORMAT(CreateDate, ‘%Y-%m’) AS month, SUM(sales) AS total_sales
FROM customer
GROUP BY DATE_FORMAT(CreateDate, ‘%Y-%m’)
ORDER BY month

Note : Why not use AS name in group by as well?. It can be used because, GROUP BY clause is performed before the select statement, so the month Alias is not yet created to be used. Hence full expression is used in the group by clause.

<u>CLASS EXERCISE</u> (Customer dataset -> PhoneNumber, Email, Address)

<u>FUNCTIONS USED IN EXERCISE</u>

SUBSTRING_INDEX(string, delimiter, count)

- Gets the substring from given string, after or before delimiter character based on count of delimiter (if count = 1 then we’ll get every substring present before delimiter, and -1 to get substring after delimiter)
- Used for removing extension in phone number standardization case, please refer the screen shots for sql queries

<u>REGULAR EXPRESSION FUNCTION</u>

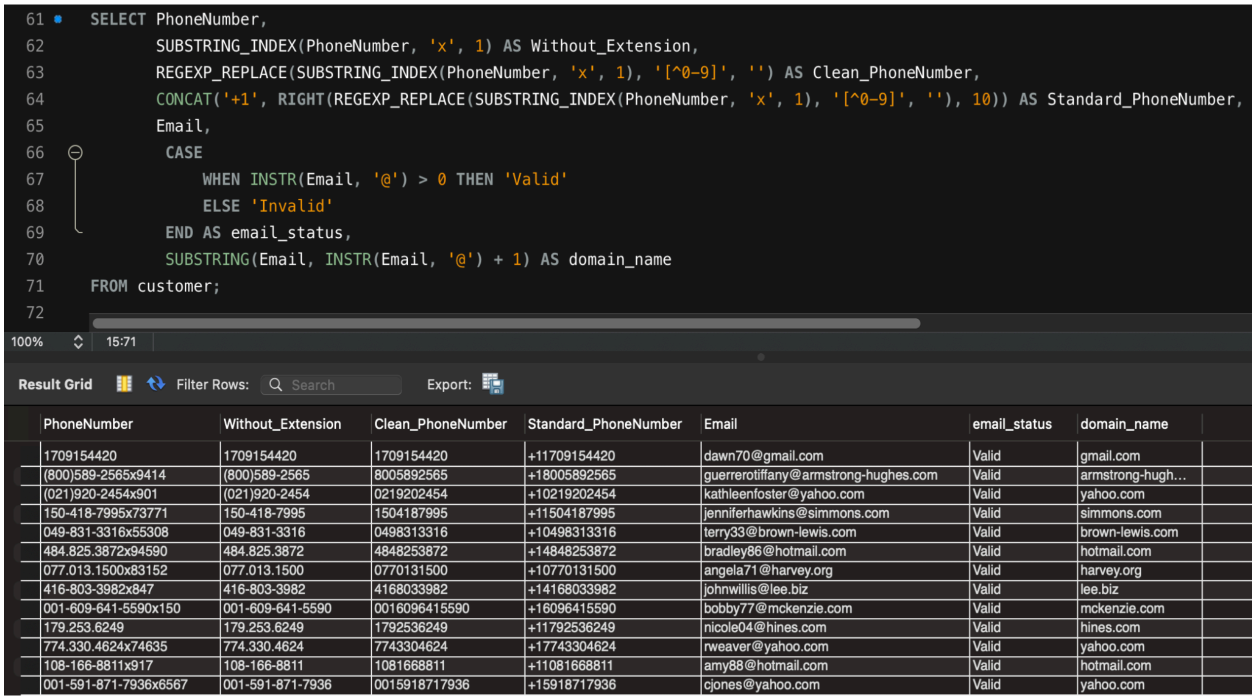
REGEXP_REPLACE(string, regular expression, replace with)

- replaces or removes (give ‘’ in replace with space)
- Used for removing unwanted characters in phone number cases. Let's say we have 72(98-xhs83. Which has non numeric chars as well.
- Define a regular expression to query only numbers using above function.

INSTR(string, substring)

- Give a string it’ll result out index position of substring present in the string
- Used in email to find @, if present its a valid email

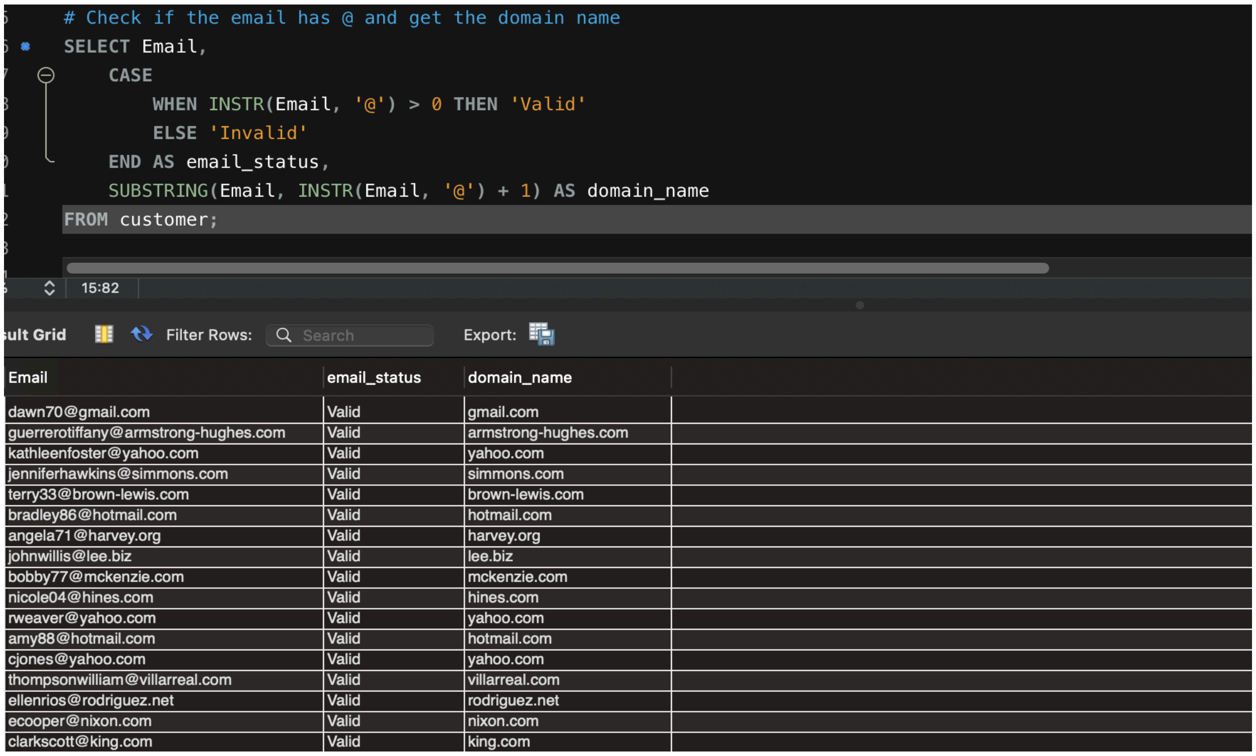
1. Standardize phone number



```sql
SELECT PhoneNumber,
       SUBSTRING_INDEX(PhoneNumber, 'x', 1) AS Without_Extension,
       REGEXP_REPLACE(SUBSTRING_INDEX(PhoneNumber, 'x', 1), '[^0-9]', '') AS Clean_PhoneNumber,
       CONCAT('+1', RIGHT(REGEXP_REPLACE(SUBSTRING_INDEX(PhoneNumber, 'x', 1), '[^0-9]', ''), 10)) AS Standard_PhoneNumber,
       Email,
        CASE
            WHEN INSTR(Email, '@') > 0 THEN 'Valid'
            ELSE 'Invalid'
        END AS email_status,
        SUBSTRING(Email, INSTR(Email, '@') + 1) AS domain_name
FROM customer;
```

PhoneNumber	Without_Extension	Clean_PhoneNumber	Standard_PhoneNumber	Email	email_status	domain_name
1709154420	1709154420	1709154420	+11709154420	dawn70@gmail.com	Valid	gmail.com
(800)589-2565x9414	(800)589-2565	8005892565	+18005892565	guerrerotiffany@armstrong-hughes.com	Valid	armstrong-hugh...
(021)920-2454x901	(021)920-2454	0219202454	+10219202454	kathleenfoster@yahoo.com	Valid	yahoo.com
150-418-7995x73771	150-418-7995	1504187995	+11504187995	jenniferhawkins@simmons.com	Valid	simmons.com
049-831-3316x55308	049-831-3316	0498313316	+10498313316	terry33@brown-lewis.com	Valid	brown-lewis.com
484.825.3872x94590	484.825.3872	4848253872	+14848253872	bradley86@hotmail.com	Valid	hotmail.com
077.013.1500x83152	077.013.1500	0770131500	+10770131500	angela71@harvey.org	Valid	harvey.org
416-803-3982x847	416-803-3982	4168033982	+14168033982	johnwillis@lee.biz	Valid	lee.biz
001-609-641-5590x150	001-609-641-5590	0016096415590	+16096415590	bobby77@mckenzie.com	Valid	mckenzie.com
179.253.6249	179.253.6249	1792536249	+11792536249	nicole04@hines.com	Valid	hines.com
774.330.4624x74635	774.330.4624	7743304624	+17743304624	rweaver@yahoo.com	Valid	yahoo.com
108-166-8811x917	108-166-8811	1081668811	+11081668811	amy88@hotmail.com	Valid	hotmail.com
001-591-871-7936x6567	001-591-871-7936	0015918717936	+15918717936	cjones@yahoo.com	Valid	yahoo.com

1. Check Email has @ and get the domain name

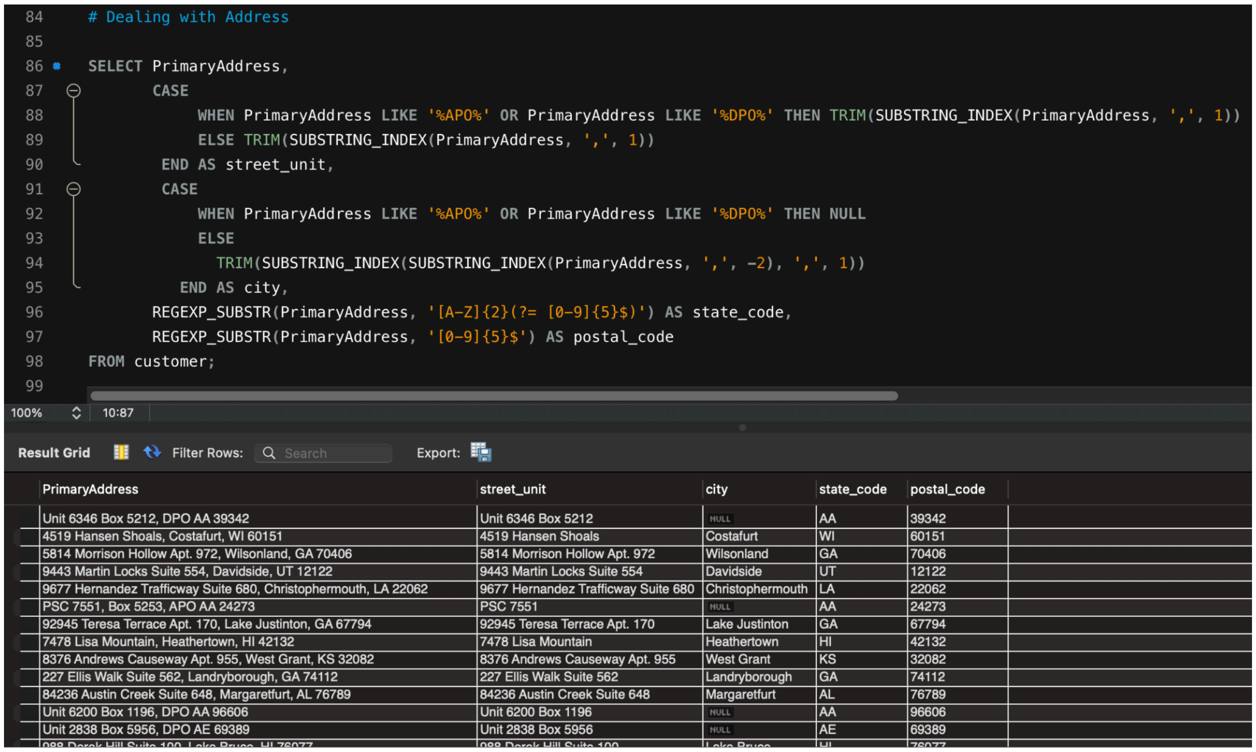


```sql
# Check if the email has @ and get the domain name
SELECT Email,
    CASE
        WHEN INSTR(Email, '@') > 0 THEN 'Valid'
        ELSE 'Invalid'
    END AS email_status,
    SUBSTRING(Email, INSTR(Email, '@') + 1) AS domain_name
FROM customer;
```

Email	email_status	domain_name
dawn70@gmail.com	Valid	gmail.com
guerrerotiffany@armstrong-hughes.com	Valid	armstrong-hughes.com
kathleenfoster@yahoo.com	Valid	yahoo.com
jenniferhawkins@simmons.com	Valid	simmons.com
terry33@brown-lewis.com	Valid	brown-lewis.com
bradley86@hotmail.com	Valid	hotmail.com
angela71@harvey.org	Valid	harvey.org
johnwillis@lee.biz	Valid	lee.biz
bobby77@mckenzie.com	Valid	mckenzie.com
nicole04@hines.com	Valid	hines.com
rweaver@yahoo.com	Valid	yahoo.com
amy88@hotmail.com	Valid	hotmail.com
cjones@yahoo.com	Valid	yahoo.com
thompsonwilliam@villarreal.com	Valid	villarreal.com
ellenrios@rodriguez.net	Valid	rodriguez.net
ecooper@nixon.com	Valid	nixon.com
clarkscott@king.com	Valid	king.com

S DHAANESH

2. Standardize Address



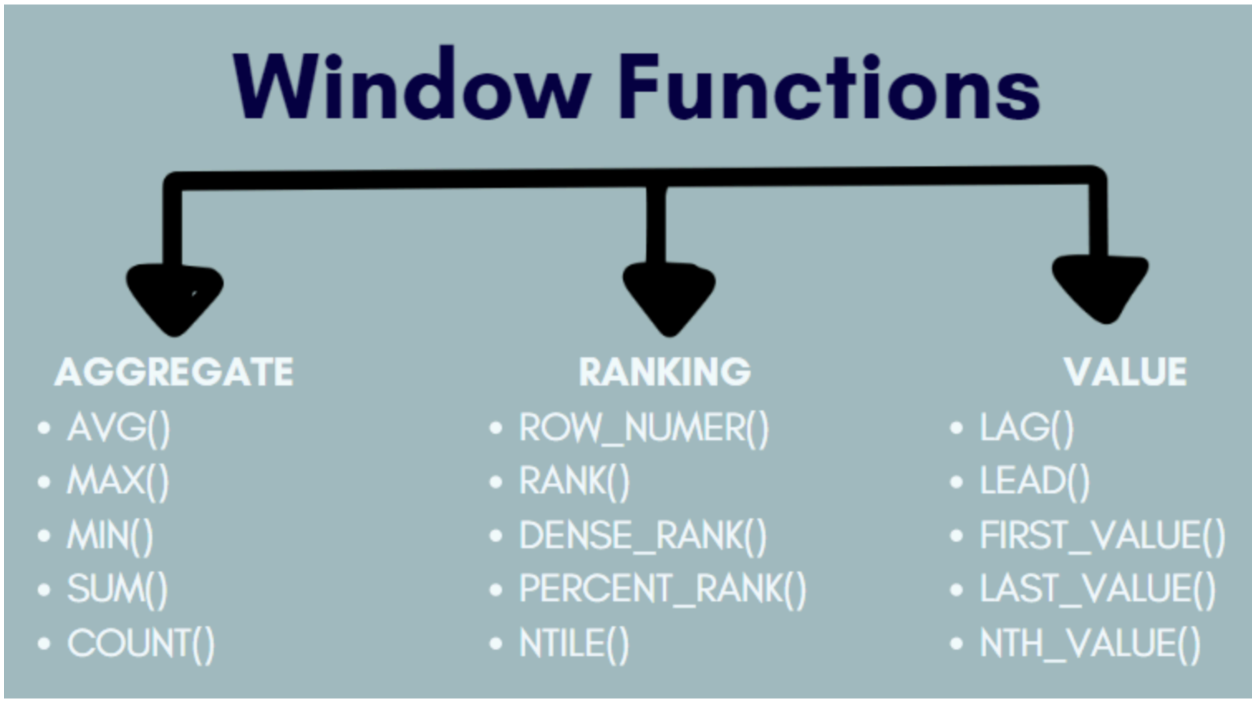
```sql
# Dealing with Address

SELECT PrimaryAddress,
        CASE
            WHEN PrimaryAddress LIKE '%APO%' OR PrimaryAddress LIKE '%DPO%' THEN TRIM(SUBSTRING_INDEX(PrimaryAddress, ',', 1))
            ELSE TRIM(SUBSTRING_INDEX(PrimaryAddress, ',', 1))
         END AS street_unit,
         CASE
            WHEN PrimaryAddress LIKE '%APO%' OR PrimaryAddress LIKE '%DPO%' THEN NULL
            ELSE
              TRIM(SUBSTRING_INDEX(SUBSTRING_INDEX(PrimaryAddress, ',', -2), ',', 1))
           END AS city,
        REGEXP_SUBSTR(PrimaryAddress, '[A-Z]{2}(?= [0-9]{5}$)') AS state_code,
        REGEXP_SUBSTR(PrimaryAddress, '[0-9]{5}$') AS postal_code
FROM customer;
```

PrimaryAddress	street_unit	city	state_code	postal_code
Unit 6346 Box 5212, DPO AA 39342	Unit 6346 Box 5212	NULL	AA	39342
4519 Hansen Shoals, Costafurt, WI 60151	4519 Hansen Shoals	Costafurt	WI	60151
5814 Morrison Hollow Apt. 972, Wilsonland, GA 70406	5814 Morrison Hollow Apt. 972	Wilsonland	GA	70406
9443 Martin Locks Suite 554, Davidside, UT 12122	9443 Martin Locks Suite 554	Davidside	UT	12122
9677 Hernandez Trafficway Suite 680, Christophermouth, LA 22062	9677 Hernandez Trafficway Suite 680	Christophermouth	LA	22062
PSC 7551, Box 5253, APO AA 24273	PSC 7551	NULL	AA	24273
92945 Teresa Terrace Apt. 170, Lake Justinton, GA 67794	92945 Teresa Terrace Apt. 170	Lake Justinton	GA	67794
7478 Lisa Mountain, Heathertown, HI 42132	7478 Lisa Mountain	Heathertown	HI	42132
8376 Andrews Causeway Apt. 955, West Grant, KS 32082	8376 Andrews Causeway Apt. 955	West Grant	KS	32082
227 Ellis Walk Suite 562, Landryborough, GA 74112	227 Ellis Walk Suite 562	Landryborough	GA	74112
84236 Austin Creek Suite 648, Margaretfurt, AL 76789	84236 Austin Creek Suite 648	Margaretfurt	AL	76789
Unit 6200 Box 1196, DPO AA 96606	Unit 6200 Box 1196	NULL	AA	96606
Unit 2838 Box 5956, DPO AE 69389	Unit 2838 Box 5956	NULL	AE	69389

THEORY

WINDOW FUNCTION

A window function performs a calculation across a set of table rows that are somehow related to the current row. unlike aggregate functions and group by.



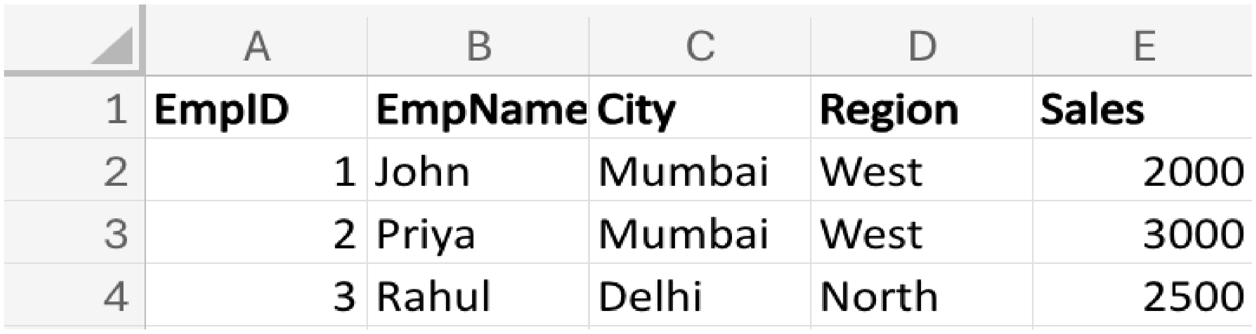
COMPONENTS

1. **PARTITION BY**

Creates a mini group within your data, if you are calculating running by total.

PARTITION BY Department - Ensures total is calculated for each department

EXAMPLE TABLE



	A	B	C	D	E
1	**EmpID**	**EmpName**	**City**	**Region**	**Sales**
2	1	John	Mumbai	West	2000
3	2	Priya	Mumbai	West	3000
4	3	Rahul	Delhi	North	2500

Q. Employee sales by city and region

S DHAANESH

```
SELECT EmpName, City, Region, Sales,
        SUM(sales) OVER (PARTITION BY City) AS total_city_sales,
        SUM(sales) OVER(PARTITION BY Region) AS total_region_sales,
        RANK() OVER(PARTITION BY City ORDER BY Sales) AS city_sales_rank
FROM employees;
```

Difference between GROUP BY & PARTITION BY

GROUP BY	**PARTITION BY**
Aggregate data into summary results	Apply window functions without collapsing
Rows are collapsed, returns one row per group	Rows are not collapsed, returns one row per row group
Aggregate functions are used (SUM, MIN, MAX, AVG)	Windows functions are used (RANK(), ROW_NUMBER()

S DHAANESH