Working with Dates in SQL in the LDP

- **Date data types**
  - **Text** - a date in Text data type may look like a date, complete with time components, but behaves as text. Usually contains a "T" before the time components. Text-type dates are present in many derived tables (LDP 1.4 or earlier). Note that the appearance may vary depending on the format of the date in the source table. **IMPORTANT:** dates extracted from a data array via "json_extract_path_text" will be in TEXT format. They must be converted to a date, timestamp or timestamptz data type to be useable in queries where you are using the date in inequalities (date A > date B), date component extractions or date calculation expressions.
    - Example: instance_ext.record_created_date: **2021-06-18T18:04:00.85**
  - **Date** - this is a simple date that contains just the core date without time components. This data type is not found natively in the LDP, but is used to convert other data types into a form that is useful for querying.
    - Example: **2022-08-15**
  - **Timestamp** - contains the date and time components, but with zeros where the timezone components would be.
    - **2021-06-25T13:10:23.509+00:00**
  - **Timestamptz** - contains the core date and time components, plus the timezone offset (-0400 or -0500 for the New York timezone; varies because of daylight savings time (0400) vs. standard time (0500))
    - **2022-05-04 14:03:23.554 -0400**
    - **2022-11-25 15:05:27.764 -0500**

- **Converting date types by "Casting"**
  - To convert a date to a different data type, use the "cast" function, represented by a double colon. Example: to cast a "timestamptz" date to a simple date, enter
    table.field_name::**DATE**

    To convert a "text" date to a timestamptz date:
    table.field_name::**TIMESTAMPTZ**

```
cl.loan_date AS raw_date,

cl.loan_date::DATE AS simple_date,

cl.loan_date::TIMESTAMP AS date_with_time,

cl.loan_date::TIMESTAMPTZ AS date_with_timezone
```

    You can do the same thing when extracting a date from a data array:

```
json_extract_path_text (ffa.data,'metadata','createdDate') as raw_date (will be TEXT data type)

json_extract_path_text (ffa.data,'metadata','createdDate')::DATE as simple_date,

json_extract_path_text (ffa.data,'metadata','createdDate')::TIMESTAMP as date_with_time,

json_extract_path_text (ffa.data,'metadata','createdDate')::TIMESTAMPTZ as date_with_timezone
```

- **Date Formatting**
  - To format a date in standard eye-readable format in your output, you may want to convert the date using the "TO_CHAR" function. Syntax is: TO_CHAR (table.field::date type, 'format that you want to see it in'). If you want to show the time components, the date has to be cast in the "**timestamp**" data type.

    TO_CHAR (cl.loan_date::**DATE**,'mm/dd/yyyy') **AS** simple_date,

    TO_CHAR (cl.loan_date::TIMESTAMP,'mm/dd/yyyy hh:mi am') AS date_with_time_and_am_pm

    TO_CHAR (json_extract_path_text (ffa.data,'metadata','createdDate')::TIMESTAMP,'mm/dd/yyyy hh:mi am') as date_with_hours_and_minutes_am_pm
  - Note that the TO_CHAR function converts the date to TEXT (it will no longer function as a date)

- **Extracting Date Parts**

  -- When you want the **month number** (1-12) or **weekday number** (1-7), use the "**Extract**" function OR the "**date_part**" function

  -- When you want the **month name** or **weekday name**, use the "**to_char**" function.

-- When you want the **numerical day of the month** (1-31), use the "**date_part**" function

```
date_part ('month', li.loan_date) as month_number_using_date_part

date_part ('day', li.loan_date) as day_of_month_number_using_date_part

date_part ('dow', li.loan_date) as day_of_week_number_using_date_part


to_char (li.loan_date, 'Mon') as month_name_using_tochar

to_char (li.loan_date, 'Dy') as day_name_using_tochar


extract (month FROM li.loan_date) as month_number_using_extract

extract (day FROM li.loan_date) as day_of_month_number_using_extract

extract (dow FROM li.loan_date) as day_of_week_number_using_extract
```

- **Finding the elapsed time between two dates - AGE, INTERVAL, simple subtraction**

  You can use the "**AGE**" function or **date subtraction** to find the time elapsed between two dates. In these example, we are using "CURRENT_DATE" as one of the dates, but you can use any two date fields, cast as the same data type.

```
AGE (CURRENT_DATE, table.field_name) AS time_elapsed

CURRENT_DATE::DATE - table.field_name::DATE AS days_elapsed
```

  - The Age function will show the results broken down into time components: Example: **10 mons 2 days 07:15:12.524**
  - The date subtraction function will show the **number of days**, if the two dates you're subtracting are cast as "DATE": Example: **309**
  - When the dates you're subtracting are in another date format (such as timestamptz), the results will show time components, which may be useful (or not). Example: **308 days 07:15:12.524**

To find the number days elapsed **since a given calendar date**, use the subtraction function:

```
current_date -'2022-07-01' as number_of_days_since_7_1_22
```

To find a **calendar date** that is some time interval from another date, use the "**INTERVAL**" function in combination with subtraction (or addition, as needed):

```
to_char ((current_date - interval '33 years 9 months 26 days')::date,'mm/dd/yyyy') AS date_record_created

    ...result displays as a date: 03/19/1989 – because that's how we formatted the expression with TO_CHAR

age (current_date,'1989-03-17') AS amount_of_time_in_collection

    ...result displays as a time interval: 33 years 9 mons 26 days
```