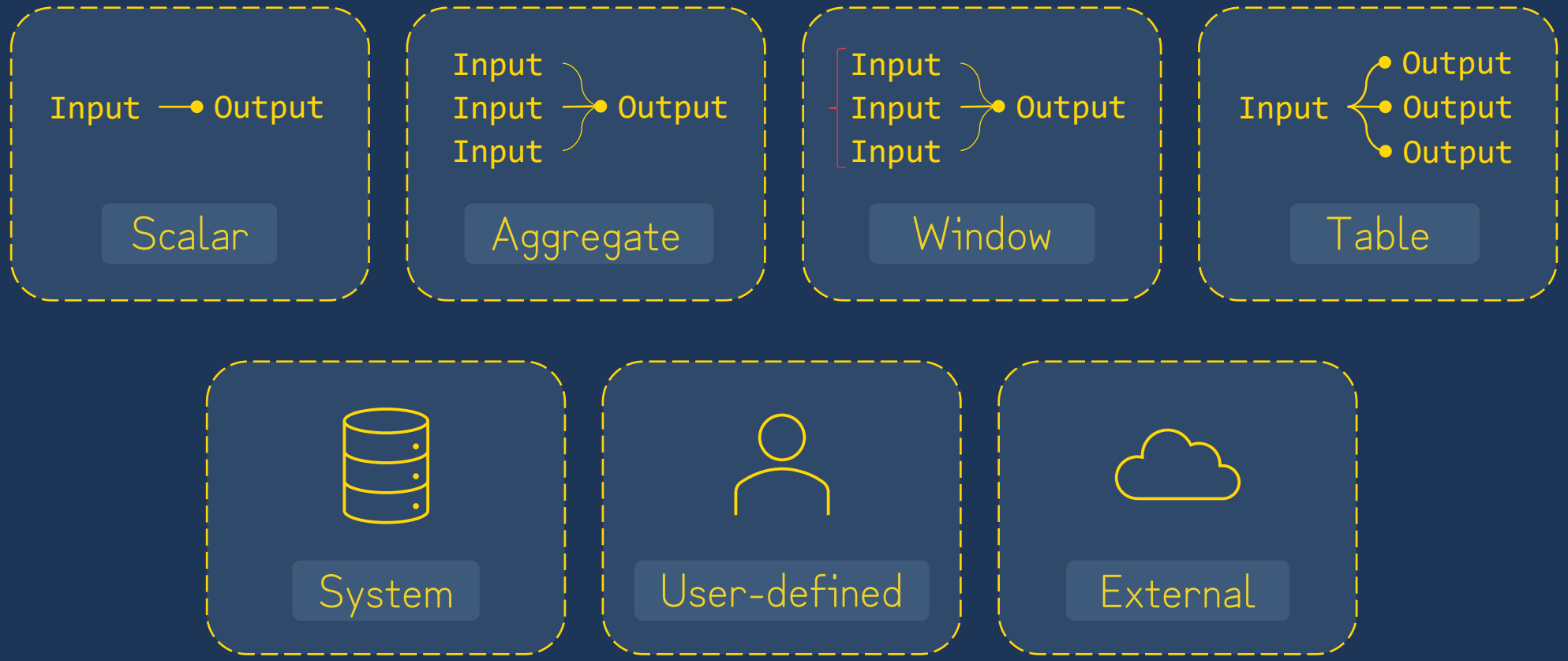
Summary of Snowflake Functions

Supported Function Types



Scalar Functions

- Bitwise Expression
- Conditional Expression
- Context
- Conversion
- Data Generation
- Date & Time
- Encryption

A **scalar function** is a function that **returns one value per invocation**; these are mostly used for returning **one value per row.**

```
SELECT UUID_STRING();
```

Output:

```
----------------------------------------
|"UUID_STRING()"                       |
----------------------------------------
|d29d4bfa-40cb-4159-9186-e10f5d59f031|
----------------------------------------
```

- Semi-structured Data
- String & Binary
- Regular Expressions
- Hash
- Metadata
- File
- Geospatial
- Numeric

Aggregate Functions

Aggregate functions operate on values across rows to perform mathematical calculations such as sum, average & counting.

- General
- Bitwise
- Boolean
- Hash
- Semi-structured
- Linear Regression

- Stats and Probability
- Distinct Values
- Cardinality Estimation
- Similarity Estimation
- Frequency Estimation
- Percentile Estimation

```
INSERT INTO ACCOUNT VALUES ('001', 10.00), ('001', 23.78),('002', 67.78);

SELECT MAX(AMOUNT) FROM ACCOUNT;
```

Output:

```
----------------
|"MAX(AMOUNT)" |
----------------
|67.78         |
----------------
```

Window Functions

Window functions are a subset of aggregate functions, allowing us to **aggregate on a subset of rows used as input to a function.**

```
SELECT ACCOUNT_ID, AMOUNT, MAX(AMOUNT) OVER (PARTITION BY ACCOUNT_ID) FROM ACCOUNT;
```

Output:

```
-----------------------------------------
|“ACCOUNT_ID” |“AMOUNT”   |“MAX(AMOUNT)” |
-----------------------------------------
|001          |10.00      |23.78         |
|001          |23.78      |23.78         |
|002          |67.78      |67.78         |
-----------------------------------------
```

Table Functions

Table functions return a **set of rows for each input row.** The returned set can contain **zero, one, or more rows.** Each row can contain **one or more columns.**

Data Loading | Data Generation | Data Conversion | Object Modelling

Semi-structured | Query Results | Usage Information

```
SELECT RANDSTR(5, RANDOM()), RANDOM() FROM TABLE(GENERATOR(ROWCOUNT => 3));
```

Output:

```
-----------------------------------------
|"RANDSTR(5,RANDOM())" |"RANDOM()"      |
-----------------------------------------
|My4FU                 |5744406107517962 11 |
|YiPSS                 |1779357660907745898|
|cu2Hw                 |6562320827285185330|
-----------------------------------------
```

System Functions

1. **System functions** provide a way to execute actions in the system.

```
SELECT system$cancel_query('01a65819-0000-2547-0000-94850008c1ee');
```

Output:

```
-----------------------------------------------------------------
|“SYSTEM$CANCEL_QUERY('01A65819-0000-2547-0000-94850008C1EE’)”|
-----------------------------------------------------------------
|query [01a65819-0000-2547-0000-94850008c1ee] terminated.     |
-----------------------------------------------------------------
```

System Functions

2 **System functions** provide information about the system.

```
SELECT system$pipe_status(‘my_pipe');
```

Output:

```
-----------------------------------------------------
|“SYSTEM$PIPE_STATUS('MYPIPE’)”                     |
-----------------------------------------------------
|{"executionState":"RUNNING","pendingFileCount":0} |
-----------------------------------------------------
```

System Functions

3 **System functions** provide information about queries.

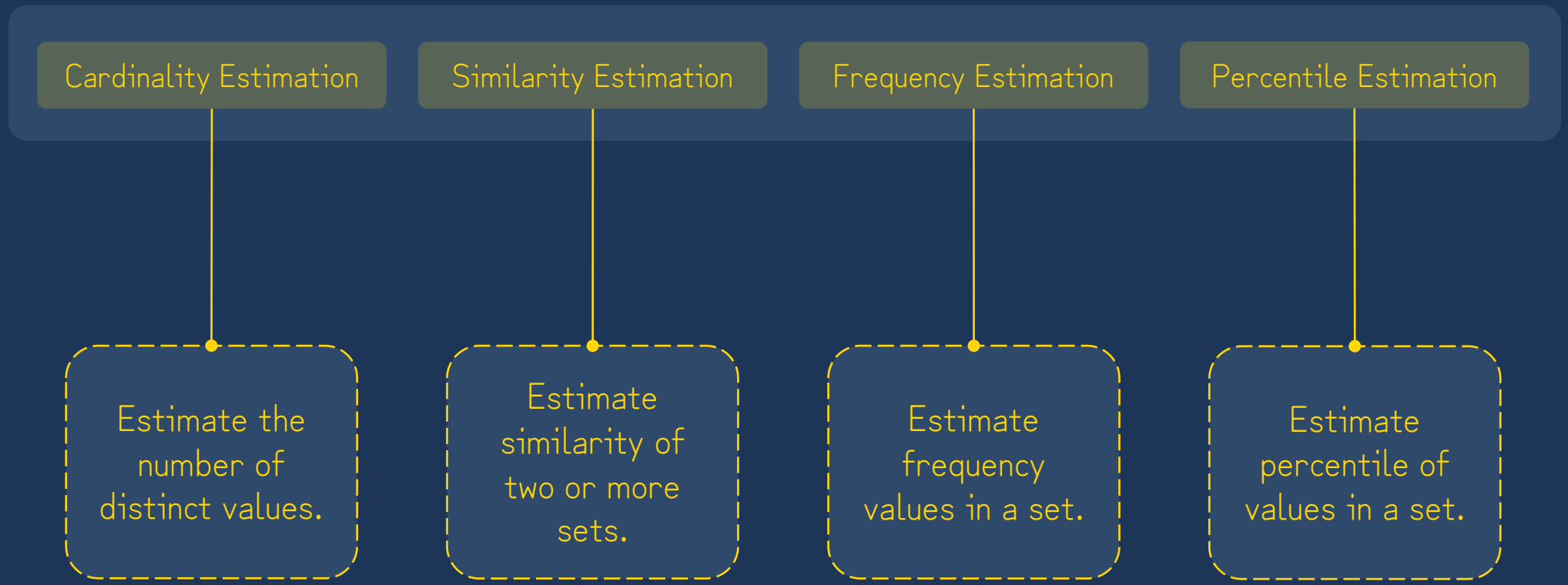
```
SELECT system$explain_plan_json('SELECT AMOUNT FROM ACCOUNT');
```

Output:

```
-------------------------------------------------------------
|“SYSTEM$EXPLAIN_PLAN_JSON('SELECT AMOUNT FROM ACCOUNT')” |
-------------------------------------------------------------
|{                                                          |
|"GlobalStats": {                                           |
|    "partitionsTotal": 1,                                  |
|    "partitionsAssigned": 1,                               |
|    "bytesAssigned": 1024                                  |
| }[…]                                                      |
-------------------------------------------------------------
```

Estimation Functions

Estimation Functions



Cardinality Estimation

Snowflake implemented something called the **HyperLogLog cardinality estimation algorithm**, which returns an **approximation of the distinct number of values** of a column.

- HLL()
- HLL_ACCUMULATE()
- HLL_COMBINE()
- HLL_ESTIMATE()
- HLL_EXPORT()
- HLL_IMPORT()

```
SELECT APPROX_COUNT_DISTINCT(L_ORDERKEY) FROM LINEITEM;
```

Output: 1,491,111,415
Execution Time: 44 Seconds

```
SELECT COUNT(DISTINCT L_ORDERKEY) FROM LINEITEM;
```

Output: 1,500,000,000
Execution Time: 4 Minutes 20 Seconds

Similarity Estimation

Snowflake have implemented a two-step process to **estimate similarity, without the need to compute the intersection or union of two sets.**

1

```
SELECT MINHASH(5, C_CUSTKEY) FROM CUSTOMER;
```

Output:

```
-------------------------
|“MINHASH(5, C_CUSTKEY) |
-------------------------
|{                      |
|  "state": [           |
|    557181968304,      |
|    67530801241,       |
|    1909814111197,     |
|    8406483771,        |
|    34962958513        |
|                       |
|  ],                   |
|  "type": "minhash",   |
|  "version": 1         |
|}                      |
-------------------------
```

Similarity Estimation

Snowflake have implemented a two-step process to **estimate similarity, without the need to compute the intersection or union of two sets.**

2

```
SELECT APPROXIMATE_SIMILARITY(MH) FROM
  (
    (SELECT MINHASH(5, C_CUSTKEY) MH FROM CUSTOMER)
       UNION
    (SELECT MINHASH(5, O_CUSTKEY) MH FROM ORDERS)
  );
```

Output:

```
-------------------------------
|"APPROXIMATE_SIMILARITY(MH)" |
-------------------------------
|0.8                          |
-------------------------------
```

Frequency Estimation

Snowflake have implemented a family of functions using the **Space-Saving algorithm** to produce an **estimation of values and their frequencies.**

- APPROX_TOP_K
- APPROX_TOP_K_ACCUMULATE
- APPROX_TOP_K_COMBINE
- APPROX_TOP_K_ESTIMATE

```
SELECT APPROX_TOP_K(P_SIZE, 3, 100000) FROM PART;
```

Output:

```
-----------------------------------------
|“APPROX_TOP_K(P_SIZE, 3, 100000)”      |
-----------------------------------------
|[[13,401087],[38,401074],[35,401033]] |
-----------------------------------------
```

Frequency Estimation

Snowflake have implemented a family of functions using the **Space-Saving algorithm** to produce an **estimation of values and their frequencies.**

APPROX_TOP_K

```
SELECT  P_SIZE, COUNT(P_SIZE) AS C FROM PART
GROUP BY P_SIZE
ORDER BY C DESC
LIMIT 3;
```

Output:

```
--------------------
|"P_SIZE" | "C"    |
--------------------
|13       |401,087 |
|38       |401,074 |
|35       |401,033 |
--------------------
```

Percentile Estimation

Snowflake have implemented the **t-Digest algorithm** as an efficient way of **estimating approximate percentile values in data sets.**

- APPROX_PERCENTILE
- APPROX_PERCENTILE_ACCUMULATE
- APPROX_PERCENTILE_COMBINE
- APPROX_PERCENTILE_ESTIMATE

```
INSERT OVERWRITE INTO TEST_SCORES VALUES (23),(67),(2),(3),(9),(19),(45),(81),(90),(11);

SELECT APPROX_PERCENTILE(score, 0.8) FROM TEST_SCORES;
```

Output:

```
-----------------------------------
|“APPROX_PERCENTILE(score,0.8)” |
-----------------------------------
|74                               |
-----------------------------------
```

Table Sampling

Table Sampling

Table sampling is a convenient way to read a **random subset of rows from a table.**

Fraction-based

```
SELECT * FROM LINEITEM  TABLESAMPLE/SAMPLE  [samplingMethod] (<probability>);
```

Row

<p>/100*n

```
SELECT * FROM LINEITEM SAMPLE BERNOULLI/ROW (50);
```

Block

```
SELECT * FROM LINEITEM SAMPLE SYSTEM/BLOCK (50);
```

```
SELECT * FROM LINEITEM SAMPLE (50) REPEATABLE/SEED (765);
```

0 to 2147483647C

Table Sampling

Table sampling is a convenient way to read a **random subset of rows from a table.**

Fixed-size

```
SELECT * FROM LINEITEM TABLESAMPLE/SAMPLE (<num> ROWS);
```

```
SELECT L_TAX, L_SHIPMODE FROM LINEITEM SAMPLE BERNOULLI/ROW (3 rows);
```

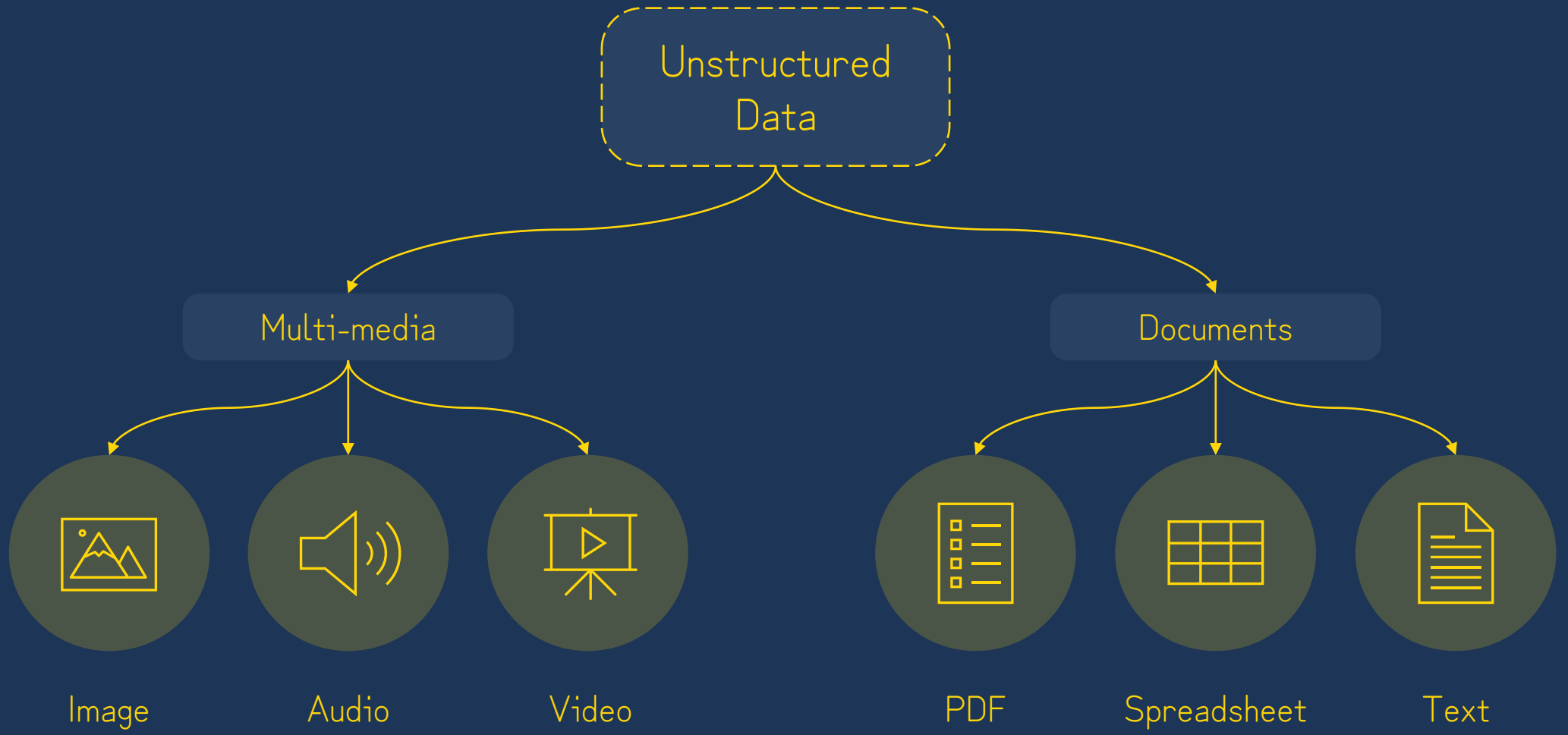
Output:

```
-------------------------
|"L_TAX" | "L_SHIPMODE" |
-------------------------
|0.02    |REG AIR       |
|0.02    |TRUCK         |
|0.06    |TRUCK         |
-------------------------
```

Fixed-size sampling does not support block sampling and use of seed. Adding these will result in an error.

Unstructured Data File Functions

Unstructured Data



File Functions

BUILD_SCOPED_FILE_URL

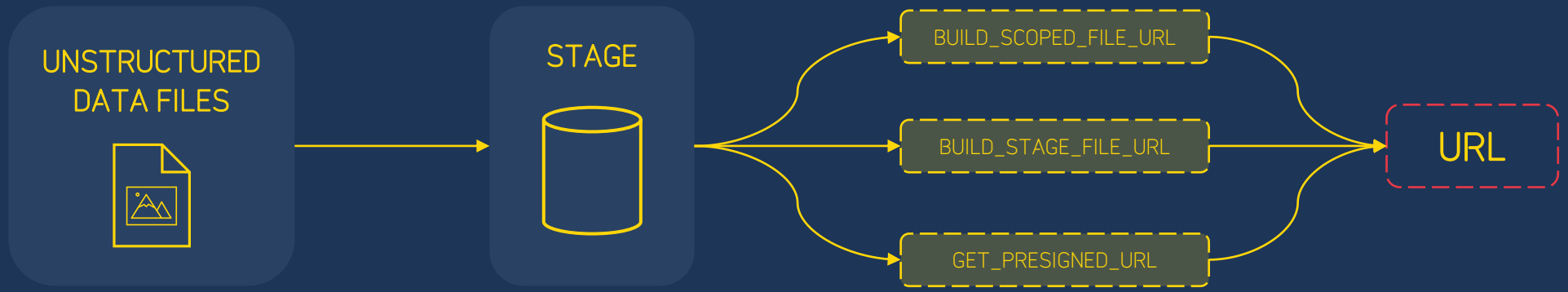
BUILD_STAGE_FILE_URL

GET_PRESIGNED_URL

GET_STAGE_LOCATION

GET_RELATIVE_PATH

GET_ABSOLUTE_PATH




```
PUT file://image.jpg @images_stage
```

```
SELECT GET_PRESIGNED_URL(@images_stage, image.jpg)
```

BUILD_SCOPED_FILE_URL

```
build_scoped_file_url( @<stage_name> , '<relative_file_path>' )
```

URL valid for 24 hours.

```
SELECT build_scoped_file_url(@images_stage, 'prod_z1c.jpg');
```

Output:

```
-----------------------------------------------------------------------
|“BUILD_SCOPED_FILE_URL(@images_stage, ‘prod_z1c.jpg’);”              |
-----------------------------------------------------------------------
|https://go44755.eu-west-2.aws.snowflakecomputing.com/api/files/      |
|01a691df-0000-277e-0000-9485000bc022/163298951696390/5fGgfDJX6kvA    |
|qZx6tUJNjWDXEu%2f8%2b7a%2fqQ5HFPCKKMs81o1MC5NSLKPzC6p2hy670VChIC7o   |
|Po2JwrY8%2fAQ13fVjwXtxs4OUf76eUDVH7G1UzOf5ugveSR6qAQF60EV7y2F9e9cn   |
|RWHBMncTyGuyCxd4gxtVSyXRQuQ7s2qBsh6%2bt0Yj4LNsOhjQFmD3EPgfGQ7P81gY   |
|z2p%2fFyRcFX4V                                                       |
-----------------------------------------------------------------------
```

When this function is called in a query the role must have USAGE privileges on an external named stage and READ privileges on an internal named stage.

BUILD_SCOPED_FILE_URL

When this function is called in a UDF, Stored Procedure or View the calling role does not require privileges on the underlying stage.

```
CREATE VIEW PRODUCT_SCOPED_URL_VIEW AS
SELECT build_scoped_file_url(@images_stage, 'prod_z1c.jpg') AS scoped_file_url;
```

```
SELECT * FROM PRODUCT_SCOPED_URL_VIEW;
```

Output:

```
-------------------------------------------------------------------------------
|SCOPED_FILE_URL                                                              |
-------------------------------------------------------------------------------
|https://go44755.eu-west-2.aws.snowflakecomputing.com/api/files/              |
|01a691df-0000-277e-0000-9485000bc022/163298951696390/5fGgfDJX6kvA            |
|qZx6tUJNjWDXEu%2f8%2b7a%2fqQ5HFPCKKMs81o1MC5NSLKPzC6p2hy670VChIC7[…]        |
-------------------------------------------------------------------------------
```

BUILD_STAGE_FILE_URL

```
build_stage_file_url( @<stage_name> , '<relative_file_path>' )
```

ⓘ The file URL does not expire.

```
SELECT build_stage_file_url(@images_stage, 'prod_z1c.jpg');
```

Output:

```
-----------------------------------------------------------------------------------------------------------------------
| “BUILD_STAGE_FILE_URL(@images_stage, 'prod_z1c.jpg’)”                                                               |
-----------------------------------------------------------------------------------------------------------------------
|https://go44755.eu-west-2.aws.snowflakecomputing.com/api/files/DEMO_DB/DEMO_SCHEMA/IMAGES_STAGE/prod_z1c.jpg |
-----------------------------------------------------------------------------------------------------------------------
```

ⓘ Calling this function whether it’s part of a query, UDF, Stored Procedure or View requires privileges on the underlying stage, that is USAGE for external stages and READ for internal stages.

```
CREATE STAGE MY_STAGE
ENCRYPTION = (TYPE = ‘SNOWFLAKE_SSE’);
```

```
ALTER STAGE MY_STAGE SET
ENCRYPTION = (TYPE = ‘SNOWFLAKE_SSE’);
```

GET_PRESIGNED_URL

```
get_presigned_url( @<stage_name> , '<relative_file_path>' , [ <expiration_time> ] )
```

```
SELECT get_presigned_url(@images_stage, 'prod_z1c.jpg’, 600);
```

Output:

```
------------------------------------------------------------------------------------------
|GET_PRESIGNED_URL(@images_stage, 'prod_z1c.jpg’,600)                                     |
------------------------------------------------------------------------------------------
|https://sfc-uk-ds1-6-customer-stage.s3.eu-west-2.amazonaws.com/vml0-s-                   |
|ukst8973/stages/42763db5-31fa-47ef-bdb1-729d81b645c2/tree.jpg?X-Amz-Algorithm=AWS4-      |
|HMAC-SHA256&X-Amz-Date=20220827T162316Z&X-Amz-SignedHeaders=host&X-Amz-Expires=599&X-    |
|Amz-Credential=AKIA4ANG2XQCHAHQKBKS%2F20220827%2Feu-west-2%2Fs3%2Faws4_request&X-Amz-    |
|Signature=0e8edf89acc24d9bd23b5387f8938671f54212be1c50c1bfe26c974ea151af55               |
------------------------------------------------------------------------------------------
```

ⓘ Calling this function whether it’s part of a query, UDF, Stored Procedure or View requires privileges on the underlying stage, that is USAGE for external stages and READ for internal stages.

GET_PRESIGNED_URL

@DOCUMENTS_STAGE

@images_stage/document.pdf

@images_stage/document_metadata.json


```
{
  “relative_path": “document.pdf",
  "author": "Corado Fernandez",
  "published_on": "2022-01-23",
  "topics":[
    "nutrition",
    "health",
    "science"
  ]
}
```

GET_PRESIGNED_URL

```
CREATE TABLE document_metadata
  (
      relative_path string,
      author string,
      published_on date,
      topics array
  );
```

```
COPY INTO document_metadata
 FROM
 (SELECT
    $1:relative_path,
    $1:author,
    $1:published_on::date,
    $1:topics
 FROM
 @documents_stage/document_metadata.json)
 FILE_FORMAT = (type = json);
```

GET_PRESIGNED_URL

```
 CREATE VIEW document_catalog AS
(
  SELECT
   author,
   published_on,
   get_presigned_url(@documents_stage, relative_path) as presigned_url,
   topics
  FROM
    documents_table
);
```

```
SELECT * FROM document_catalog;
```

Output:

```
-------------------------------------------------------------------------------------------------------------------
|AUTHOR           |PUBLISHED_ON |PRESIGNED_URL                              |TOPICS                              |
-------------------------------------------------------------------------------------------------------------------
|Corado Fernandez |2022-01-23   |https://sfc-uk-ds1-6-customer-stage.[…] |["nutrition","health","science"] |
-------------------------------------------------------------------------------------------------------------------
```

Directory Tables

Directory Tables

```
CREATE STAGE INT_STAGE
DIRECTORY = (ENABLE = TRUE)
```

Internal

Directory Table

External

Directory Table

```
ALTER STAGE EXT_STAGE SET
DIRECTORY = (ENABLE = TRUE)
```

```
SELECT * FROM DIRECTORY(@INT_STAGE)
```

```
-----------------------------------------------------------------------------------------------------------------------
|RELATIVE_PATH |SIZE    |LAST_MODIFIED |MD5          |ETAG         |FILE_URL                                          |
-----------------------------------------------------------------------------------------------------------------------
|document.pdf  |250,838 |55:42.0       |ba247312[…] |f76b4327[…] |https://go44755.eu-west-2.aws.snowflake[…] |
-----------------------------------------------------------------------------------------------------------------------
```

Refreshing Directory Tables

Directory Tables must be refreshed to reflect the most up-to-date changes made to stage contents. This includes **new files uploaded, removed files and changes to files in the path.**

Internal

External

```
ALTER STAGE INT_STAGE REFRESH;
```

Refreshing Directory Tables

Directory Tables must be refreshed to reflect the most up-to-date changes made to stage contents. This includes **new files uploaded, removed files and changes to files in the path.**

External

1. Enable a directory table on an external stage.

```
CREATE STAGE INT_STAGE
DIRECTORY = (ENABLE = TRUE)
```

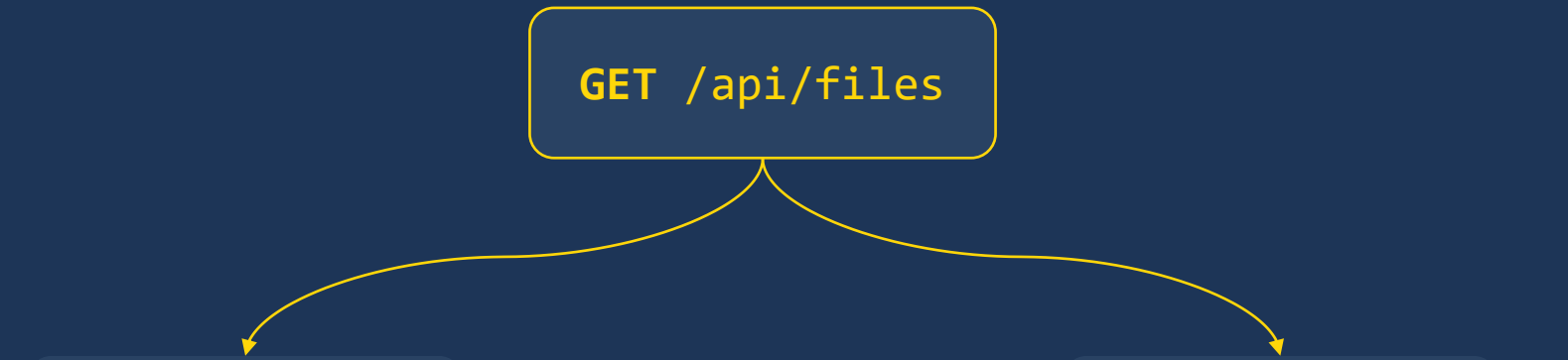
2. Describe stage and retrieve ARN for Snowflake managed SQS Queue in field **directory_notification_channel.**

```
DESCRIBE STAGE EXT_STAGE;
```

3. Configure event notifications for a S3 bucket to notify Snowflake managed SQS queue associated with the stage when new or updated data is available to read into the directory table metadata.

File Support REST API

File Support REST API




```
BUILD_SCOPED_FILE_URL();
```

```
BUILD_STAGE_FILE_URL();
```

Authorization

Only the user who generated the scoped URL can download the staged file.

Any role that has privileges on the underlying stage can access the file.

File Support REST API

```
1   import requests
2   response = requests.get(“https://go44755.eu-west-2.aws.snowflakecomputing.com/api/files/DB/SCHEMA/STAGE/img.jpg”,
3       headers={
4         "User-Agent": "reg-tests",
5         "Accept": "*/*",
6         "X-Snowflake-Authorization-Token-Type": "OAUTH",
7         "Authorization": """Bearer {}""".format(token)
8         },
9       allow_redirects=True)
10  print(response.status_code)
11  print(response.content)
```