Lesson 1: Quick Start   7 of 7 points

Go through the Colab Notebook, run all cells, complete all the exercises and answer the
questions.

✓ **What is dlt?** *   1/1

○ An open-source Python library for data transforming

◉ An open-source Python library for data loading   ✓

○ An open-source Python platform for data management

○ A library for delta live tables

✓ **Which tasks does dlt NOT automate?** *   1/1

○ Schema creation

○ Data normalization

○ Incremental loading

◉ Data modeling   ✓

✓ **Why would a Data Team use dlt? (multi-select)** *   1/1

☑ To focus on more valuable tasks   ✓

☑ Reduce costs   ✓

☑ For data security (avoid third-party clouds)   ✓

☑ For data governance   ✓

✓ Where can you run a dlt pipeline? *   1/1

○ You can run it only in serverless functions as a webhook

○ Only on the local machine

◉ Wherever Python runs   ✓

○ dlt provides its own cloud runner

✓ How does dlt handle database schemas? *   1/1

○ Produces different schemas on different DB systems.

◉ Produces the same schemas on different DB systems with slightly different   ✓
   datatypes based on database support.

○ Does not produce schemas.

○ Only works with duckdb.

✓ **Exercise 1**   *2/2

   Fetch the data from the **pokemon** table into a dataframe and count the
   number of columns in the table:

○ 3

○ 4

◉ 6   ✓

○ I don't know

Lesson 2: dlt sources and resources. Create first dlt pipeline   7 of 7 points

Go through the [Colab Notebook](#), run all cells, complete all the exercises and answer the

questions.

✓ **What is one of the main benefits of using a resource in the dlt library? ***   1/1

○ It allows you to group related resources into a single source.

○ It allows you to transform data from a resource.

⦿ It allows you to extract data from a specific endpoint or data source, and can   ✓
   also perform transformations on the data before yielding it.

✓ **What is the primary role of a source in the dlt library and why is it useful? ***   1/1

○ The primary role of a source is to yield data from a specific endpoint or data
   source. It is useful for data extraction.

○ The primary role of a source is to transform data from a resource. It is useful for
   data transformation.

⦿ The primary role of a source is to group related resources into a single source.   ✓
   It is useful for promoting modularity and organization in your codebase.

✓ **What is the main difference between a resource and a transformer in the   *** 1/1
   dlt library?

○ A resource yields data from a specific endpoint or data source, while a transformer
   returns one or more resources.

○ A resource returns one or more resources, while a transformer yields data from a
   specific endpoint or data source.

⦿ A resource yields data from a specific endpoint or data source, while a   ✓
   transformer transforms data from a resource.

✓ **Exercise 1:** Create a pipeline for GitHub API - repos endpoint. *   2/2

How many columns does the **github_repos** table have?

○ 6

◉ 106   ✓

○ 206

○ I don't know

✓ **Exercise 2:** Create a pipeline for GitHub API - stargazers endpoint. *   2/2

How many columns does the **github_stargazer** table have?

◉ 21   ✓

○ 24

○ 44

○ 124

○ I don't know

Lesson 3: Pagination & Authentication & dlt Configuration   6 of 6 points

Go through the [Colab Notebook,](#) run all cells, complete all the exercises and answer the
questions.

✓ What features does the RestAPIClient support? *　　　1/1

○ Automatic pagination

○ Various authentication mechanisms

○ Customizable request/response handling

◉ All of the above　　　　　　　　　　　　　　　✓

✓ What are the three ways to set up configurations and credentials in dlt? *　　1/1

○ ENVs

○ In code `dlt.secrets` and `dlt.config`

○ In files `secrets.toml` and `config.toml`

◉ All of the above　　　　　　　　　　　　　　　✓

✓ Exercise 1: Pagination with RESTClient　　*　　2/2

What type of pagination do we use for GitHub API?

○ Json link paginator

◉ Header link paginator　　　　　　　　　　　　✓

○ Offset paginator

○ I don't know

✓   **Exercise 2: Run pipeline with dlt.secrets.value** *   2/2

Who has id=17202864 in the stargazers table?

○ sh-rp

○ burnash

◉ rudolfix   ✓

○ I don't know

Lesson 4: Using pre-build sources and destinations   13 of 13 points

Go through the Colab Notebook, run all cells, complete all the exercises and answer the
questions.

✓   What is a verified source in dlt? *   1/1

○ A source that requires user-defined transformations

◉ A pre-built, fully customizable source for data pipelines   ✓

○ A debugging tool for checking pipeline logs

○ A built-in destination for storing pipeline data

✓ Which command initializes a dlt project with a verified source and a   *1/1
destination?

○ dlt start <source> <destination>

○ dlt build <source> <destination>

◉ dlt init <verified-source> <destination>   ✓

○ dlt source <verified-source> <destination>

✓ What is the main advantage of using the rest_api built-in source in dlt? *   1/1

○ It directly loads data into DuckDB without transformations.

◉ It provides a declarative way to configure REST API sources, handling   ✓
pagination and authentication automatically.

○ It enables automatic schema inference for SQL databases.

○ It eliminates the need for access tokens in REST API calls.

✓ What is the difference between RestAPI Client and rest_api_source *   1/1

○ The RestAPI Client is used for authentication, while rest_api_source is used for
pagination.

○ The RestAPI Client and rest_api_source are the same thing in the dlt library.

○ The RestAPI Client is used to define the configuration object, while rest_api_source
is used to connect to the API's endpoints.

◉ The RestAPI Client is used by rest_api_source under the hood, rest_api_source   ✓
has a declarative interface for creating Rest API dlt sources.

✓   Which names below are dlt **destinations**? *   1/1

○   Postgres, Github, Stripe

○   Streamlit, Google Colab, Duckdb

⦿   Postgres, Duckdb, BigQuery   ✓

○   dlt has only custom destinations

✓   **Exercise 1: Run rest_api source**   *   2/2

How many columns does the `issues` table have?

○   18

⦿   128   ✓

○   345

○   I don't know

✓   **Exercise 2: Create dlt source with rest_api**   *   2/2

How many columns does the `contributors` table have?

⦿   22   ✓

○   144

○   512

○   I don't know

✓ **Exercise 3: Run sql_database source**　　　*　　　2/2

How many columns does the `family` table have?

○ 3

○ 17

◉ 37　　　　　　　　　　　　　　　　　　　✓

○ I don't know

✓ **Exercise 4: Run filesystem source**　　　*　　　2/2

How many columns does the `userdata` table have?

◉ 15　　　　　　　　　　　　　　　　　　　✓

○ 35

○ 85

○ I don't know

Lesson 5: Write Disposition and incremental loading　　　7 of 8 points

Go through the Colab Notebook, run all cells, complete all the exercises and answer the questions.

✓ **What is the purpose of incremental loading in data pipelines?** *   1/1

○ To execute a full refresh of the dataset every time the pipeline runs, disregarding any changes or updates made to the source data since the last run.

◉ To load only new or changed data instead of reloading the entire dataset. This   ✓ approach provides several benefits, including low-latency data transfer and cost savings.

○ To purge the existing data in the destination dataset and load the new data, potentially leading to data loss if not managed properly.

○ None of the above options.

✗ **What is the best way to deal with loading stateless (unchangeable) data?** * 0/1

◉ Via replace write disposition   ✗

○ Via upsert write disposition

○ Via append write disposition

○ Via merge write disposition

✓ **What are the three options available for the write_disposition parameter** *1/1 **in dlt for incremental loading?**

○ load, append, merge

◉ replace, append, merge   ✓

○ replace, add, merge

○ load, add, upsert

✓ What is the default write_disposition in dlt? *

○ Replace

○ Merge

◉ Append ✓

○ Incremental

1/1

✓ What must you specify when using the merge write disposition in dlt? *

○ Schema name

◉ Primary key or merge key ✓

○ File loader format

○ Incremental field

1/1

✓ Which dlt function is used to define an incremental field for tracking new or updated data?

○ dlt.incremental_load

○ dlt.pipeline.track

◉ dlt.sources.incremental ✓

○ dlt.data.cursor

*1/1

✓ **Exercise 1: Make the GitHub API pipeline incremental** *   2/2

How many columns does the `comments` table have?

○ 5

○ 15

○ 55

◉ 58   ✓

○ I don't know

Lesson 6: How dlt works   7 of 7 points

Go through the [Colab Notebook](#), run all cells, complete all the exercises and answer the questions.

✓ What are the **three main steps** in the dlt process? *   1/1

◉ Extract, Normalize, and Load   ✓

○ Extract, Transform, and Load

○ Extract, Normalize, and Transfer

○ Extract, Transform, and Transfer

✓ What is the primary purpose of the **Extract** step in a data pipeline? *   1/1

○ To move data to its final destination

○ To convert raw data into meaningful insights

⦿ To retrieve raw data from various sources   ✓

○ To archive old data for backup purposes

✓ What does the **Normalize** step in the dlt process do? *   1/1

○ It loads the data into the destination

⦿ It unpacks the nested structure into relational tables, infers data types, links   ✓
  tables to create parent-child relationships, and creates a schema

○ It requests data from a source like an API

○ It ensures data safely ends up in the destination

✓ What is the primary function of the **Load** step in a data pipeline? *   1/1

○ To retrieve data from source systems

○ To apply data cleaning and transformation rules

⦿ To move data into a target system or storage destination   ✓

○ To visualize the data for analysis

✓ What is one benefit of using **progress="log"** in a pipeline configuration? *   1/1

○ It speeds up the extraction process

◉ It provides real-time metrics on CPU and memory usage   ✓

○ It compresses intermediary files to save space

○ It enables automatic retries in case of failures

✓ How can you specify the file format for intermediary files directly in the   *1/1
pipeline.run() method?

○ Using intermediary_file_format

○ Using loader_format

○ Using file_format

◉ Using loader_file_format   ✓

✓ What does the **file_max_items** parameter control in a pipeline? *   1/1

○ The maximum size of the in-memory buffer

○ The number of intermediary files created during extraction or normalization

◉ The maximum number of items stored in a single intermediary file   ✓

○ The number of files that can be loaded simultaneously

Lesson 7: Inspecting & Adjusting Schema   7 of 7 points

Go through the Colab Notebook, run all cells, complete all the exercises and answer the
questions.

✓   **What does the schema describe in the context of dlt? ***   1/1

◯   The pipeline configuration

◯   The data normalization settings

◉   The structure of normalized data, such as tables and columns   ✓

◯   The CLI commands

✓   **What does the version_hash in a schema represent? ***   1/1

◯   The schema's creation date

◯   The schema's name

◉   A hash generated from the schema's content   ✓

◯   The destination database name

✓   **How does the dlt library infer the initial schema for a pipeline run? ***   1/1

◯   It uses a predefined schema

◯   It uses the schema from the previous pipeline run

◉   It automatically infers the schema from the first pipeline run   ✓

◯   It does not infer a schema

✓   What is the default **naming** convention used by dlt? *   1/1

○   CamelCase

◉   snake_case   ✓

○   kebab-case

○   PascalCase

✓   Where can you view **schemas** in dlt? *   1/1

○   Through load packages

○   Import files

○   Export files

◉   All of the above   ✓

✓   **Exercise 1: Adjust import schema**   *   2/2

    What **data type** does the column 'version' in the **_dlt_version** table have?

◉   bigint   ✓

○   float

○   text

○   I don't know

Lesson 8: Understanding Pipeline Metadata   5 of 6 points

Go through the Colab Notebook, run all cells, complete all the exercises and answer the questions.

✓ What is a **load package** in dlt? * 1/1

◉ A collection of jobs with data for specific tables generated during a pipeline execution ✓

◯ A module for transforming data

◯ A configuration file for managing source credentials

◯ A summary of pipeline errors

✓ What is a **state** in the context of the dlt library? * 1/1

◉ A Python dictionary that lives alongside your data. ✓

◯ A Python list that stores all your data.

◯ A Python function that processes your data.

✓ What is one of the main uses of the **state** in the dlt library? * 1/1

◉ To store and retrieve values across pipeline runs, useful for implementing incremental loading and avoiding duplicate requests. ✓

◯ To store and retrieve values within a single pipeline run, useful for data processing and transformation.

◯ To store and retrieve values from external databases, useful for data backup and recovery.

✓ **What does the pipeline's last run trace contain in dlt?** *   1/1

○ It contains information about the last run of the pipeline, including loading data.

○ It contains information about the current pipeline run, including the date of the last data.

◉ It contains information about the last run of the pipeline, including things like row counts.   ✓

✗ **How can you view the most recent load package using CLI?** *   0/1

○ dlt pipeline -v <pipeline_name> schema-info

○ dlt pipeline <pipeline_name> trace

○ dlt pipeline <pipeline_name> load-package

◉ dlt pipeline -v <pipeline_name> info   ✗

✓ **What is the _dlt_loads table used for?** *   1/1

◉ Tracking completed loads with their statuses   ✓

○ Storing all pipeline configuration details

○ Archiving historical schemas

○ Visualizing pipeline statistics

Google Forms