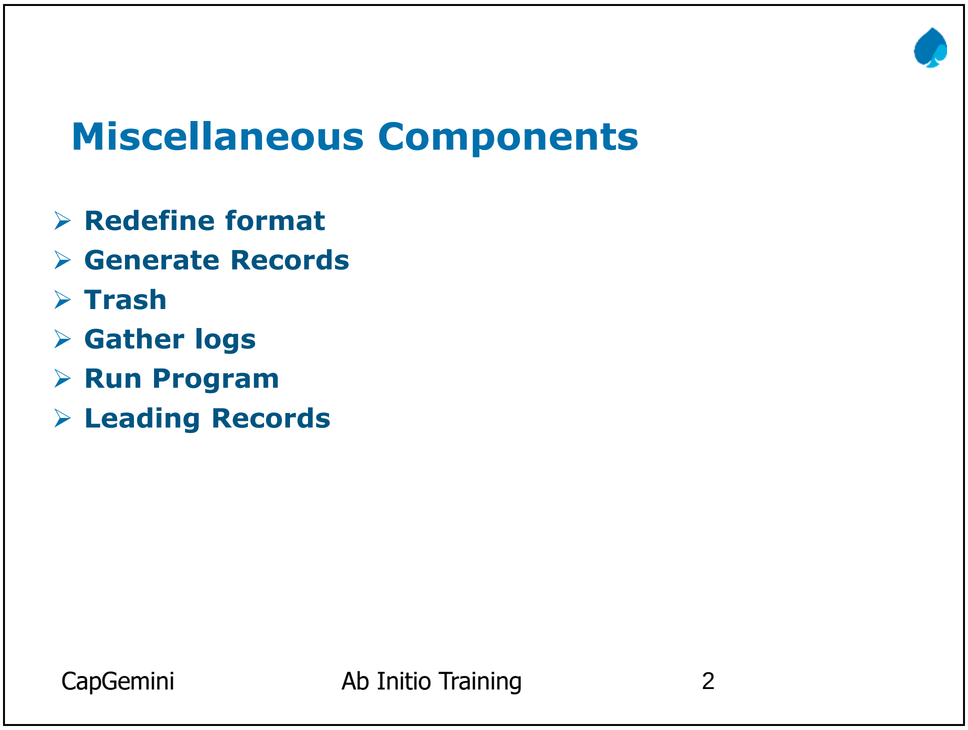
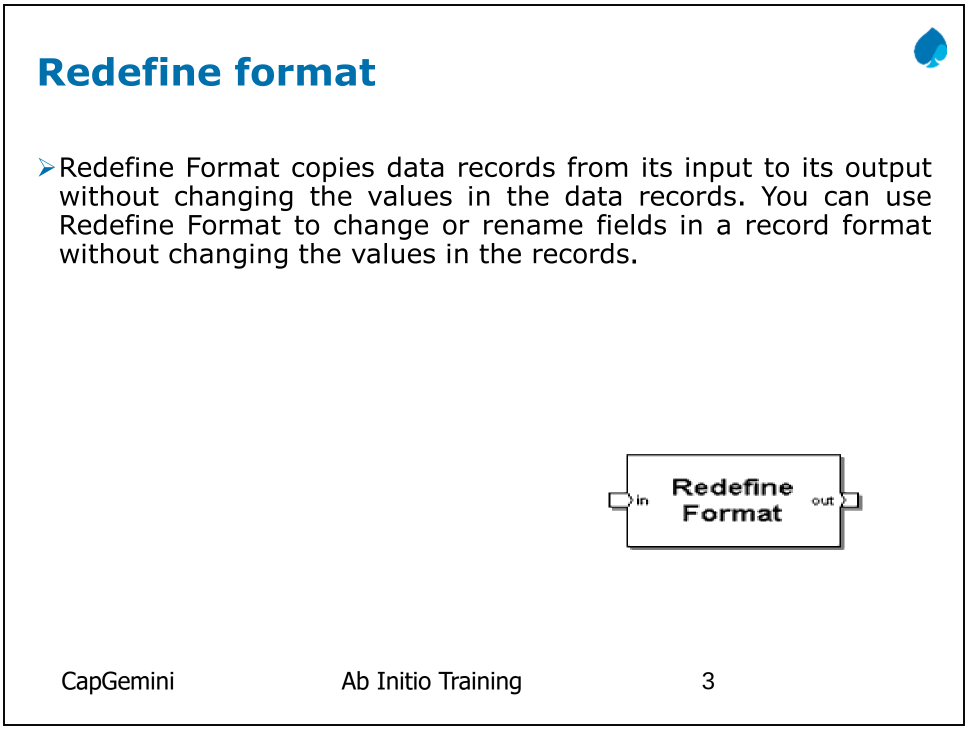
Ab initio Session 11

Introduction to Ab Initio



Miscellaneous Components

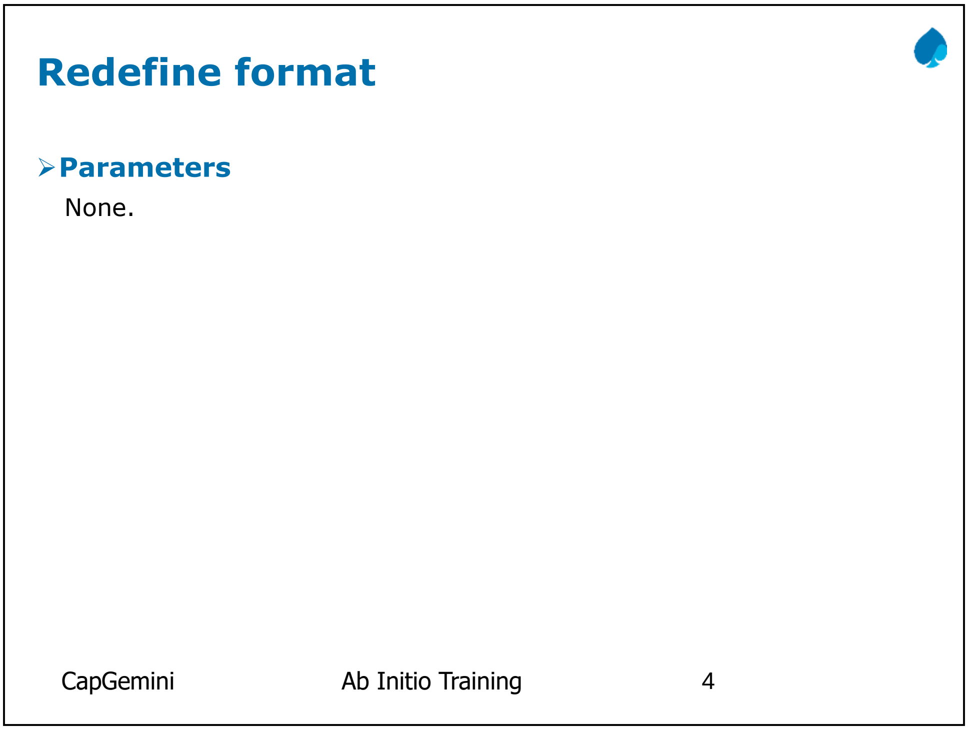
- Redefine format
- Generate Records
- Trash
- Gather logs
- Run Program
- Leading Records



Redefine format

- Redefine Format copies data records from its input to its output without changing the values in the data records. You can use Redefine Format to change or rename fields in a record format without changing the values in the records.

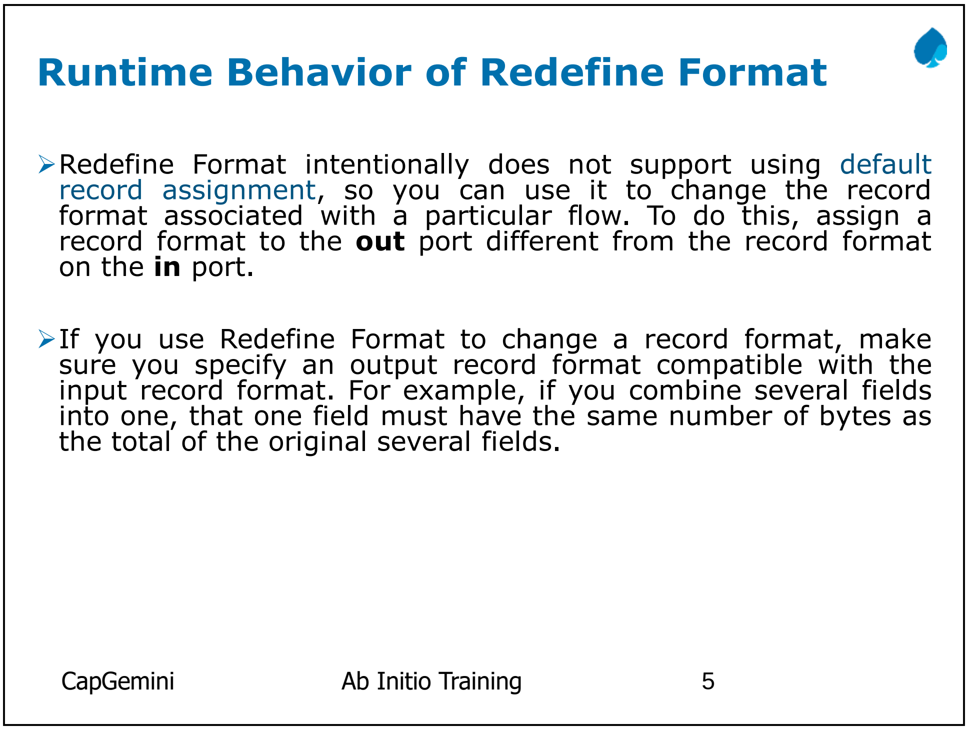
in
Redefine Format
out



Redefine format

- **Parameters**

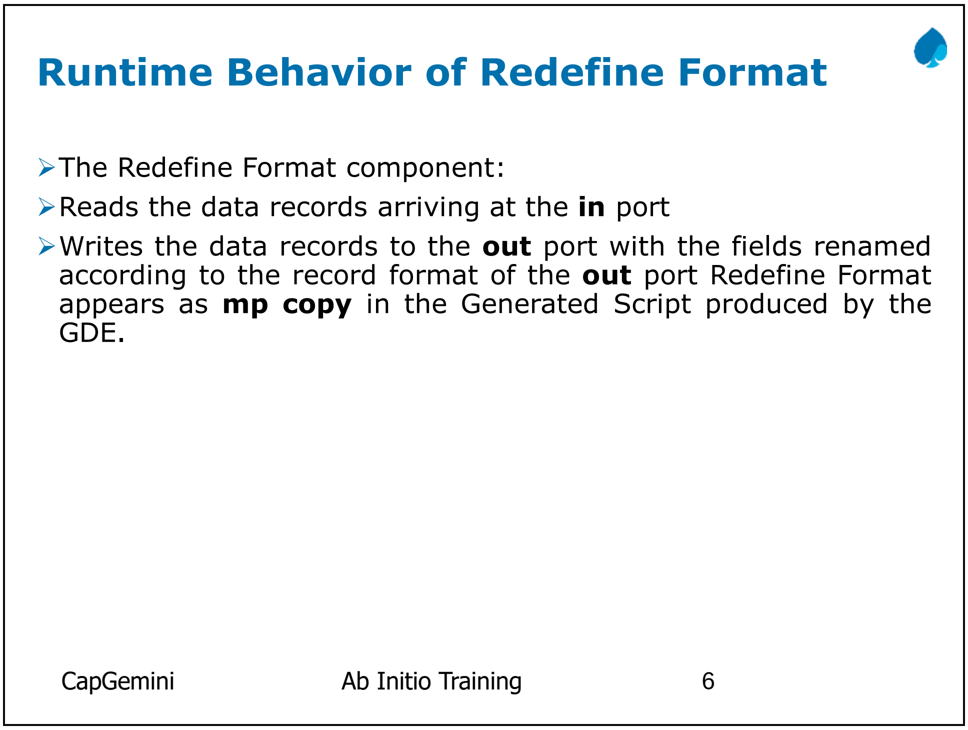
  None.



Runtime Behavior of Redefine Format

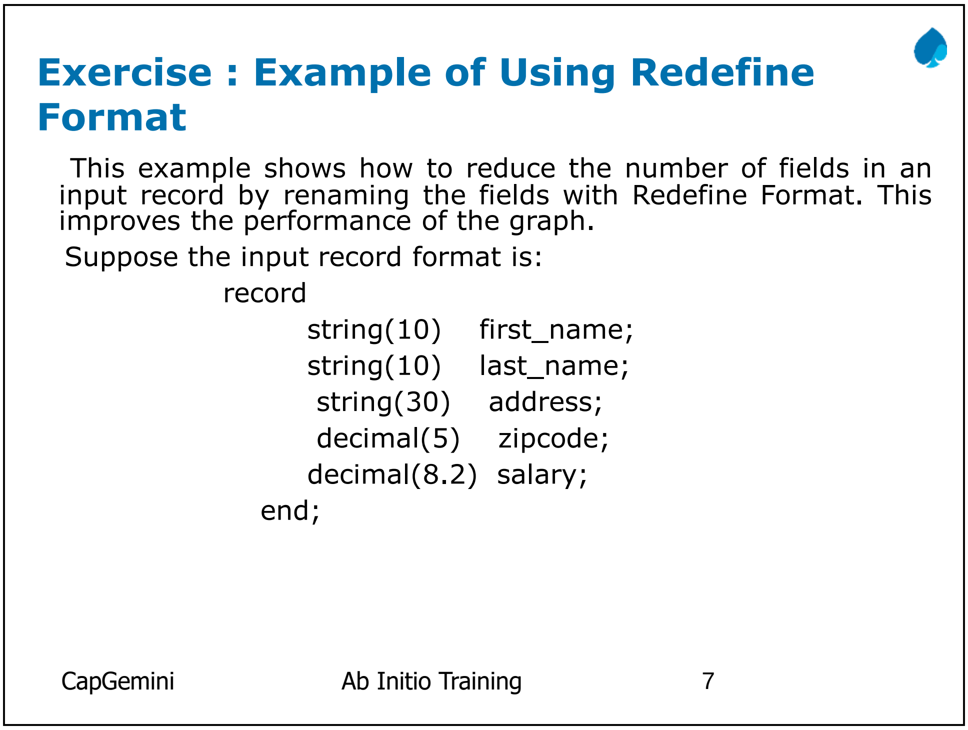
- Redefine Format intentionally does not support using default record assignment, so you can use it to change the record format associated with a particular flow. To do this, assign a record format to the **out** port different from the record format on the **in** port.

- If you use Redefine Format to change a record format, make sure you specify an output record format compatible with the input record format. For example, if you combine several fields into one, that one field must have the same number of bytes as the total of the original several fields.



Runtime Behavior of Redefine Format

- The Redefine Format component:
- Reads the data records arriving at the **in** port
- Writes the data records to the **out** port with the fields renamed according to the record format of the **out** port Redefine Format appears as **mp copy** in the Generated Script produced by the GDE.

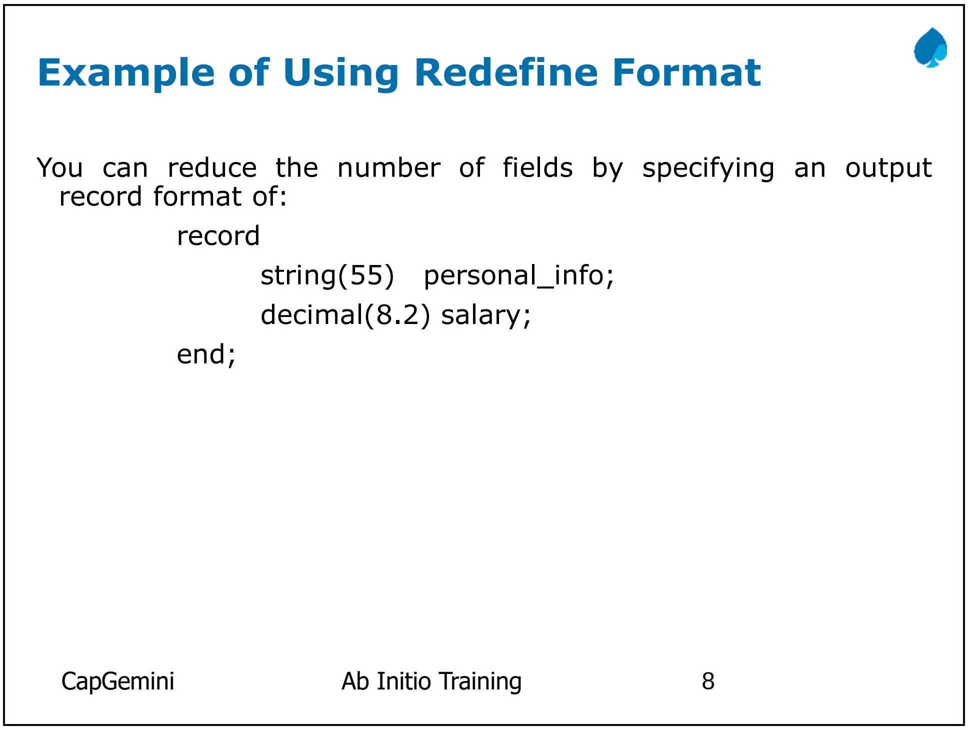


Exercise : Example of Using Redefine Format

This example shows how to reduce the number of fields in an input record by renaming the fields with Redefine Format. This improves the performance of the graph.

Suppose the input record format is:

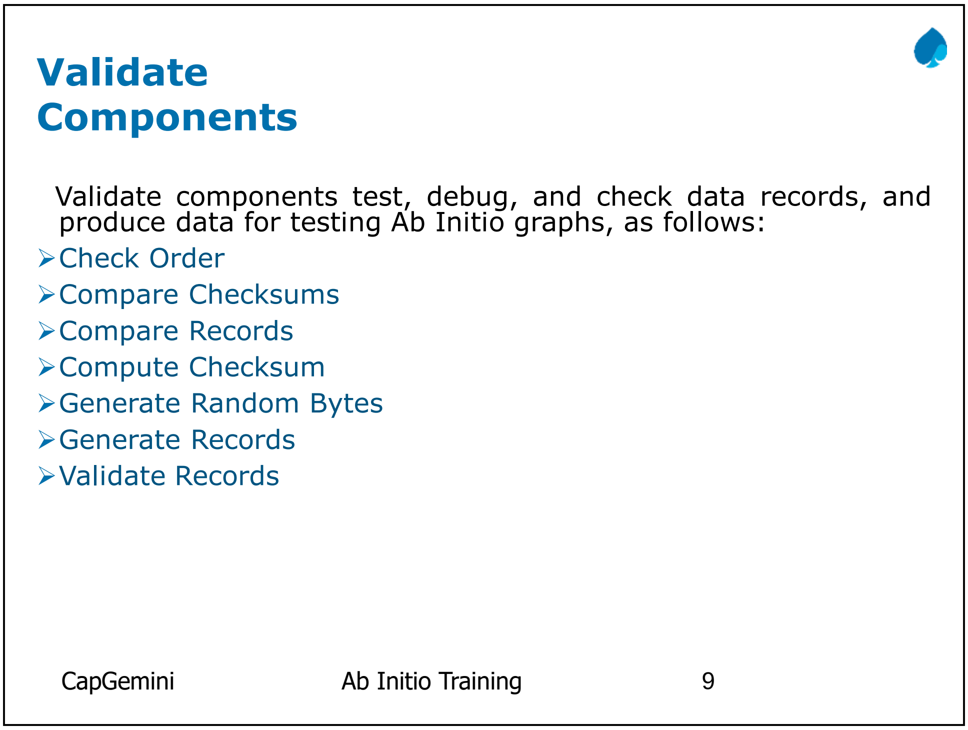
```
record
        string(10)    first_name;
        string(10)    last_name;
         string(30)    address;
         decimal(5)    zipcode;
        decimal(8.2)  salary;
    end;
```



Example of Using Redefine Format

You can reduce the number of fields by specifying an output record format of:

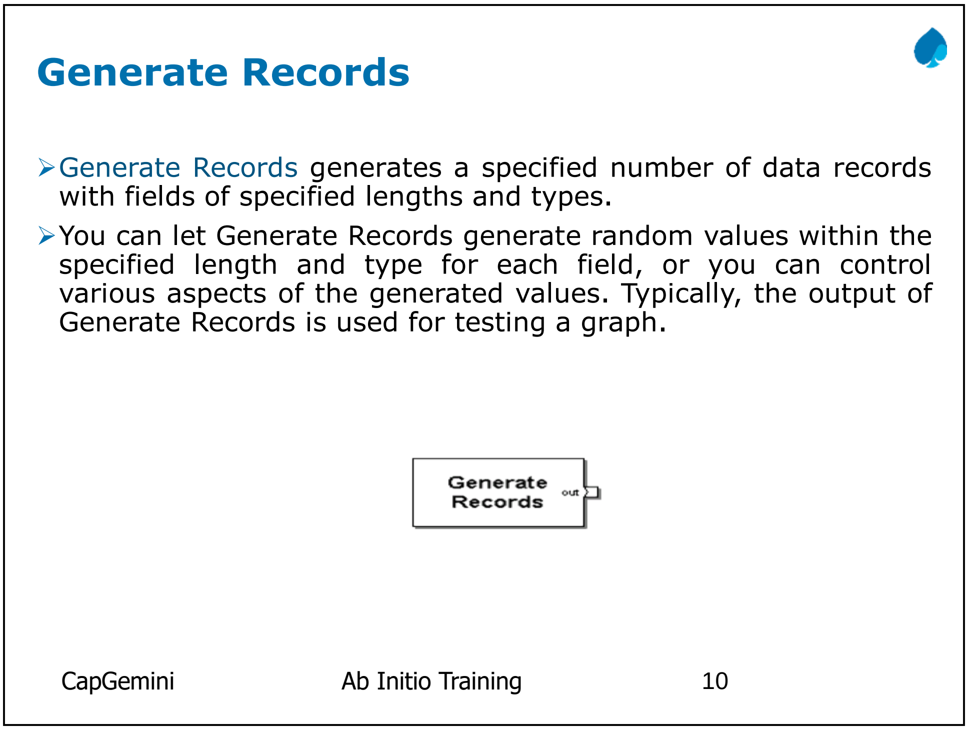
```
record
        string(55)   personal_info;
        decimal(8.2) salary;
end;
```



Validate Components

Validate components test, debug, and check data records, and produce data for testing Ab Initio graphs, as follows:

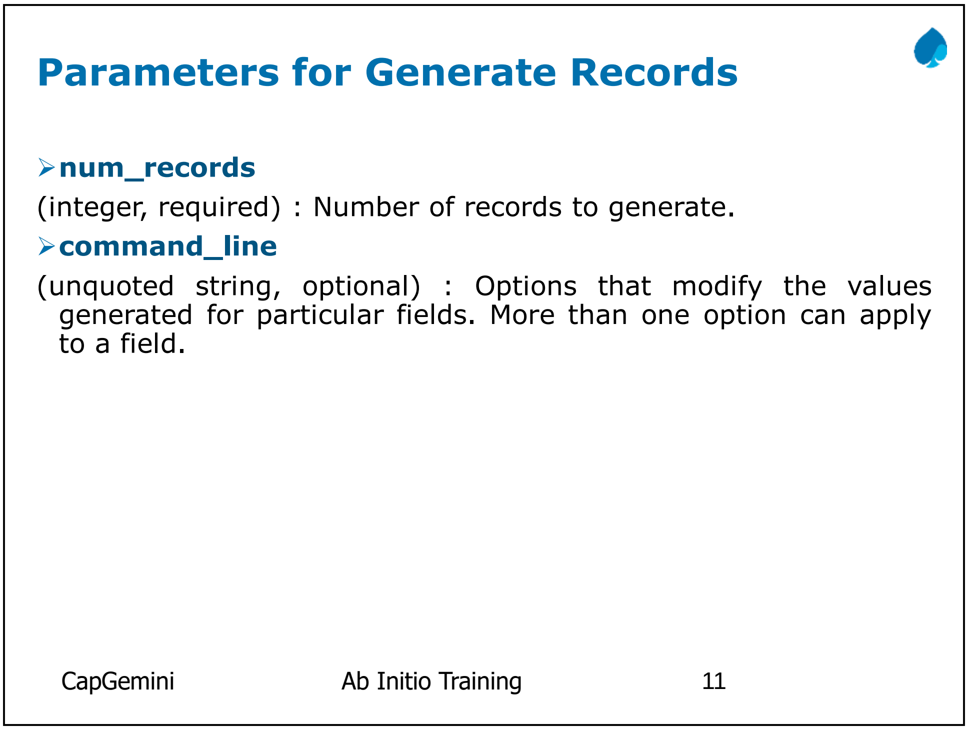
- Check Order
- Compare Checksums
- Compare Records
- Compute Checksum
- Generate Random Bytes
- Generate Records
- Validate Records



Generate Records

- Generate Records generates a specified number of data records with fields of specified lengths and types.
- You can let Generate Records generate random values within the specified length and type for each field, or you can control various aspects of the generated values. Typically, the output of Generate Records is used for testing a graph.

Generate Records
out



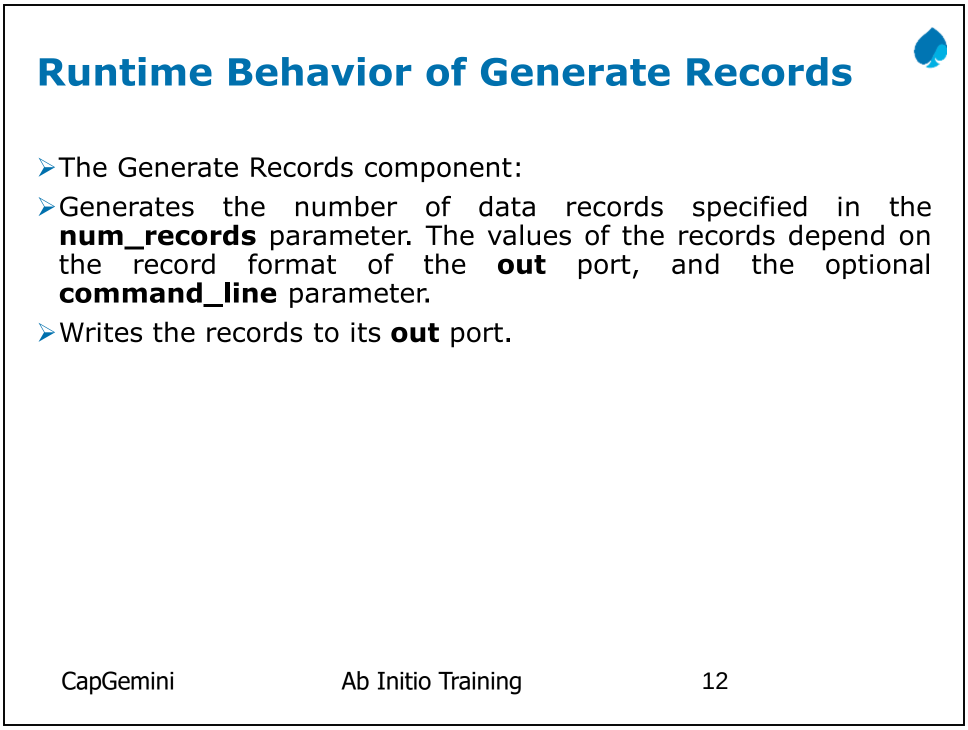
Parameters for Generate Records

- **num_records**

(integer, required) : Number of records to generate.

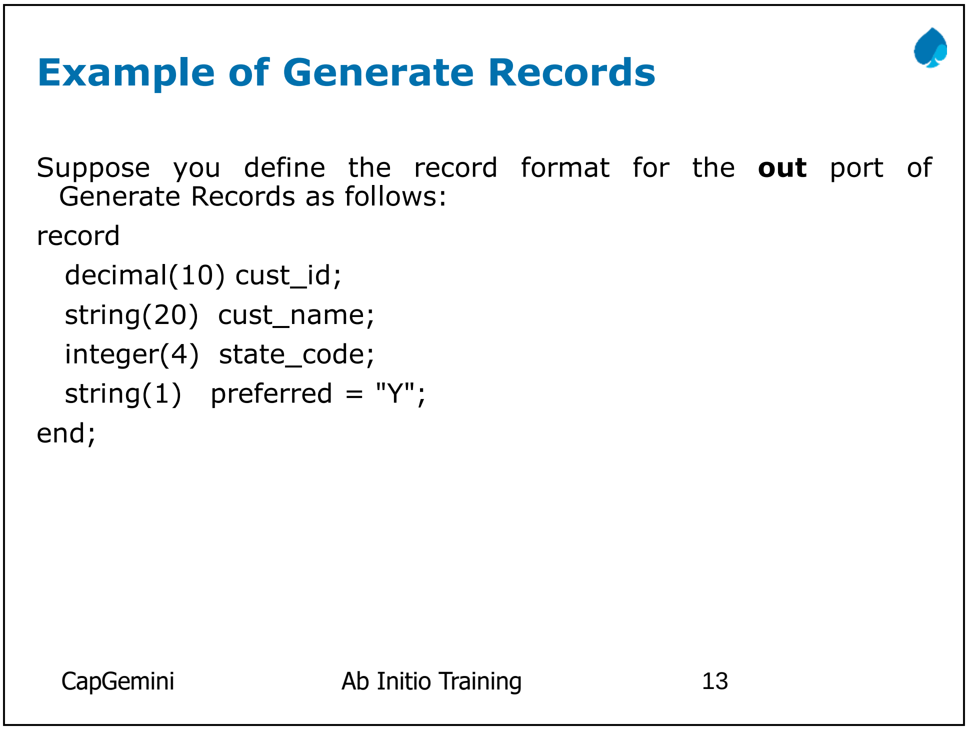
- **command_line**

(unquoted string, optional) : Options that modify the values generated for particular fields. More than one option can apply to a field.



Runtime Behavior of Generate Records

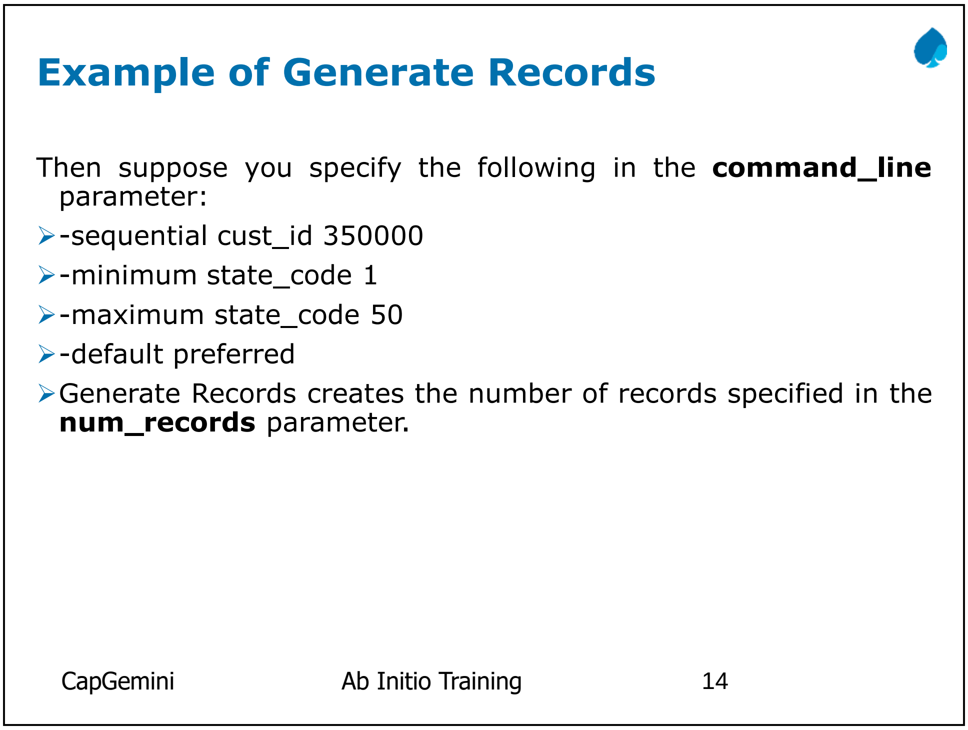
- The Generate Records component:
- Generates the number of data records specified in the **num_records** parameter. The values of the records depend on the record format of the **out** port, and the optional **command_line** parameter.
- Writes the records to its **out** port.



Example of Generate Records

Suppose you define the record format for the **out** port of Generate Records as follows:

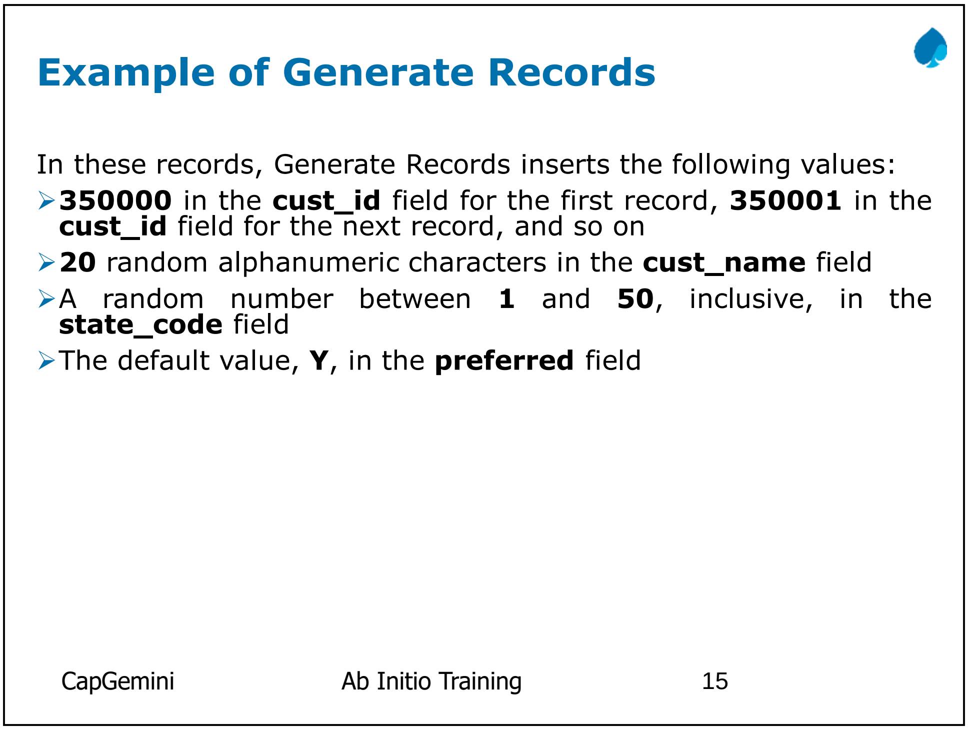
```
record
   decimal(10) cust_id;
   string(20)  cust_name;
   integer(4)  state_code;
   string(1)   preferred = "Y";
end;
```



Example of Generate Records

Then suppose you specify the following in the **command_line** parameter:

- -sequential cust_id 350000
- -minimum state_code 1
- -maximum state_code 50
- -default preferred
- Generate Records creates the number of records specified in the **num_records** parameter.



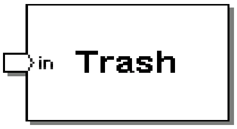
Example of Generate Records

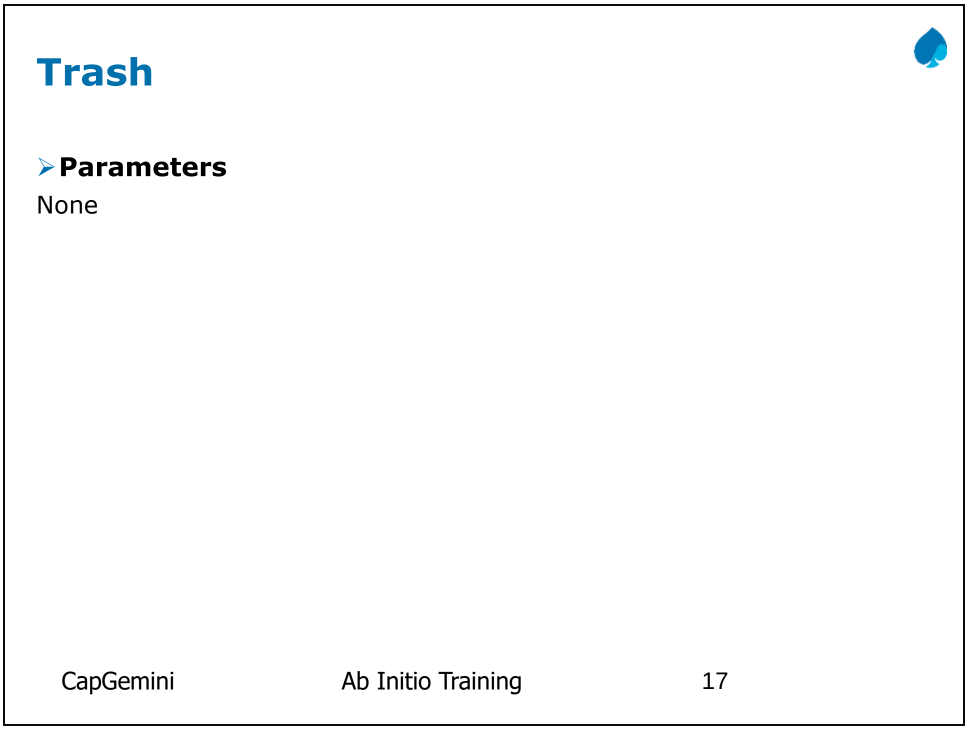
In these records, Generate Records inserts the following values:

- **350000** in the **cust_id** field for the first record, **350001** in the **cust_id** field for the next record, and so on
- **20** random alphanumeric characters in the **cust_name** field
- A random number between **1** and **50**, inclusive, in the **state_code** field
- The default value, **Y**, in the **preferred** field

Trash

- Trash ends a flow by accepting all the data records in it and discarding them.



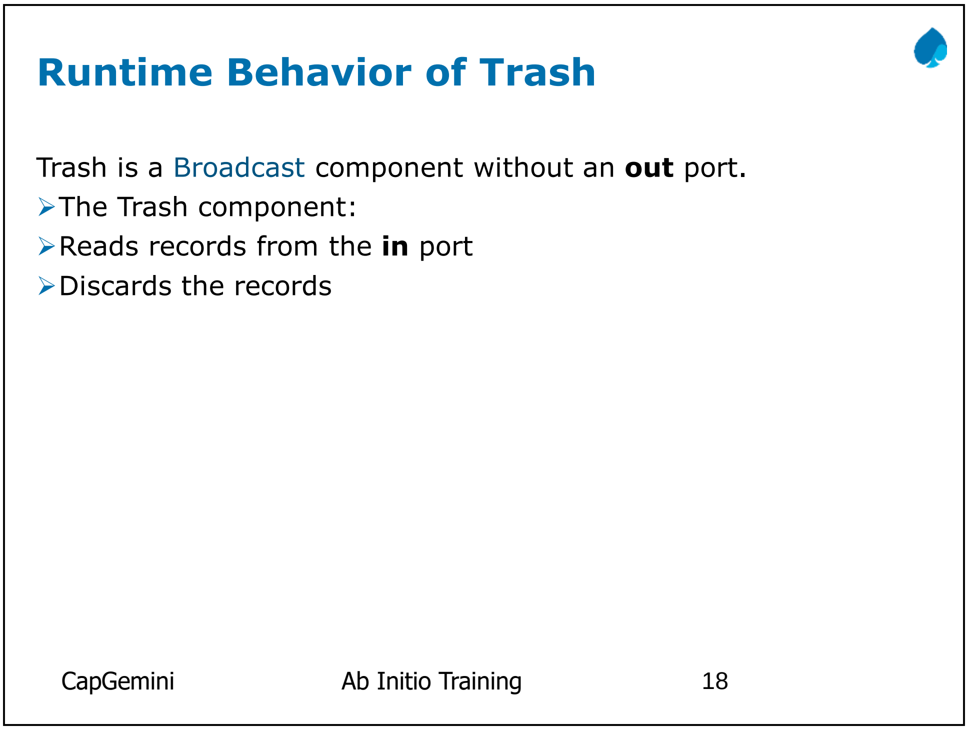


Trash

- **Parameters**

None

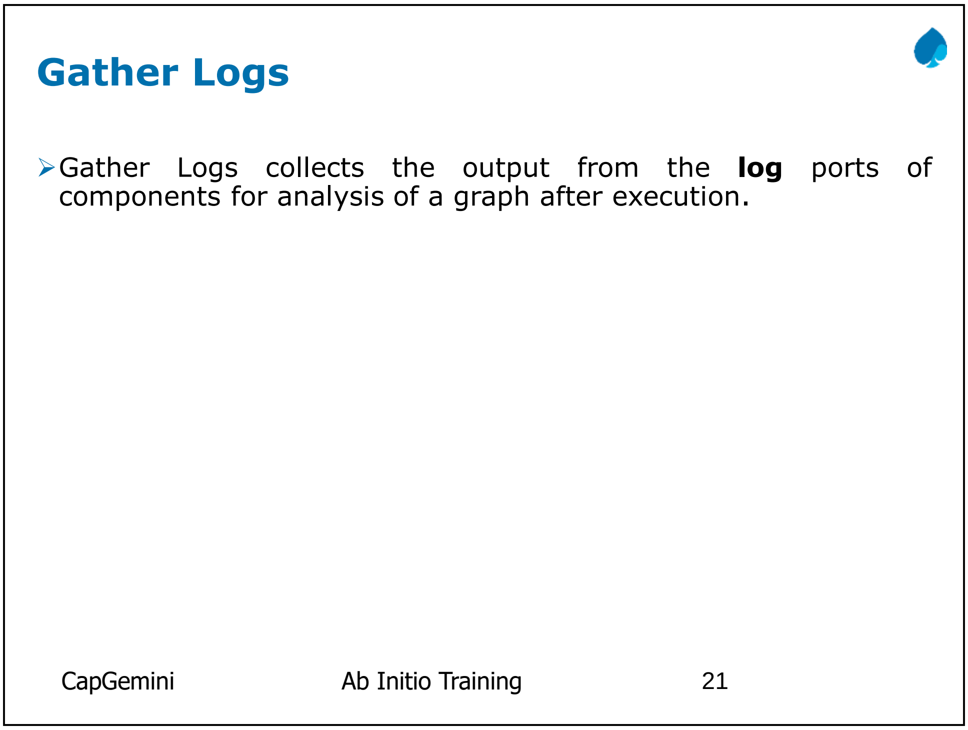
CapGemini Ab Initio Training 17



Runtime Behavior of Trash

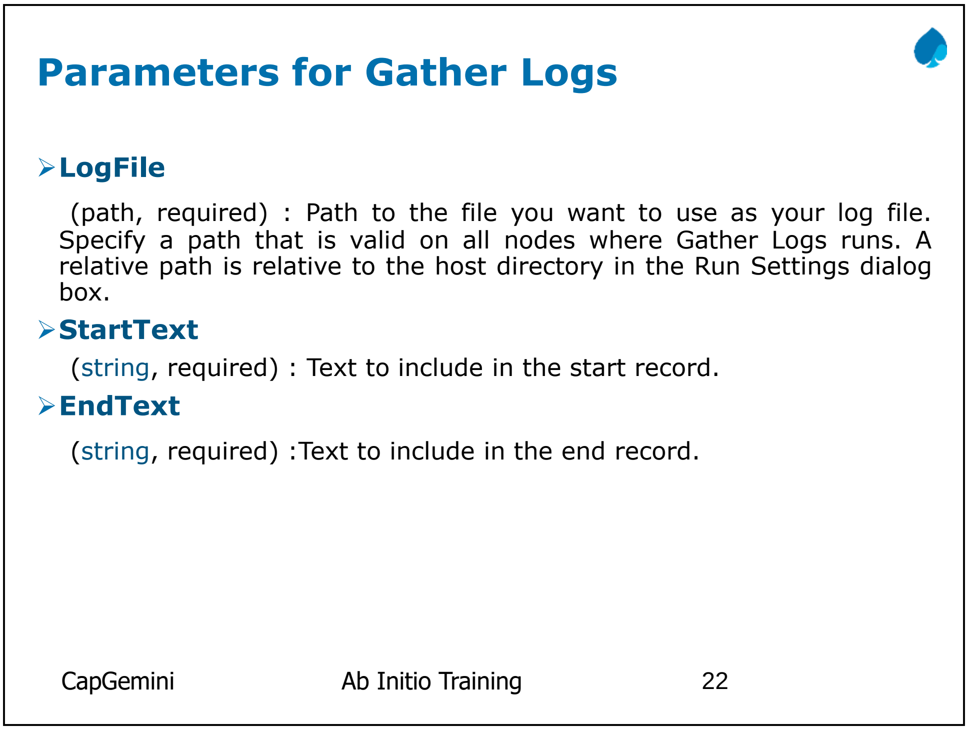
Trash is a Broadcast component without an **out** port.

- The Trash component:
- Reads records from the **in** port
- Discards the records



Gather Logs

- Gather Logs collects the output from the **log** ports of components for analysis of a graph after execution.



Parameters for Gather Logs

- **LogFile**

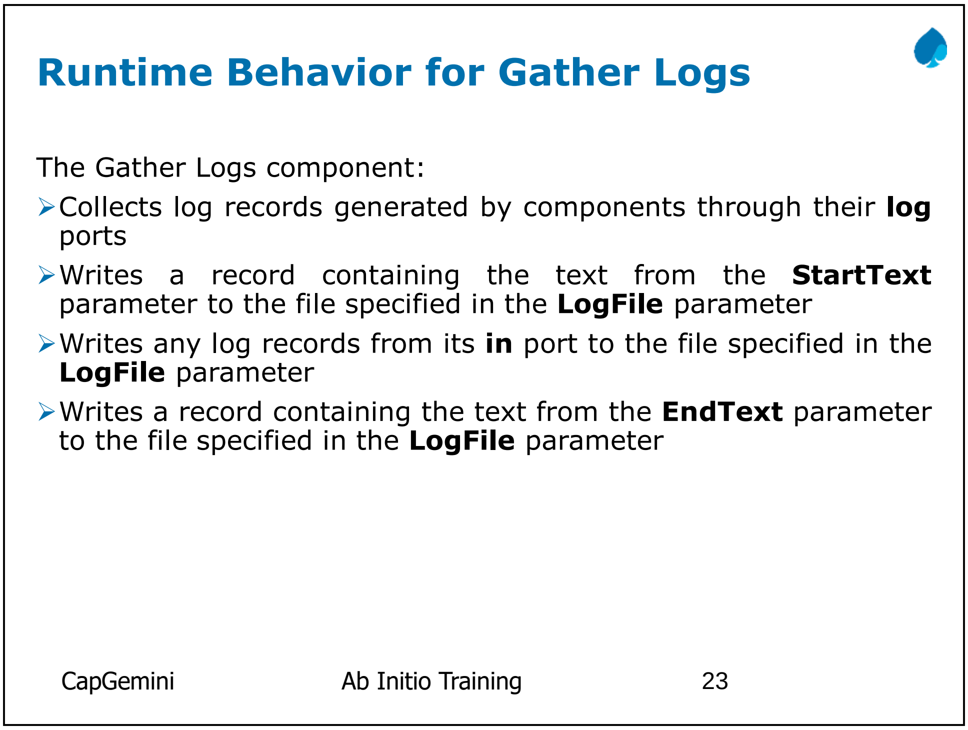
  (path, required) : Path to the file you want to use as your log file. Specify a path that is valid on all nodes where Gather Logs runs. A relative path is relative to the host directory in the Run Settings dialog box.

- **StartText**

  (string, required) : Text to include in the start record.

- **EndText**

  (string, required) :Text to include in the end record.



Runtime Behavior for Gather Logs

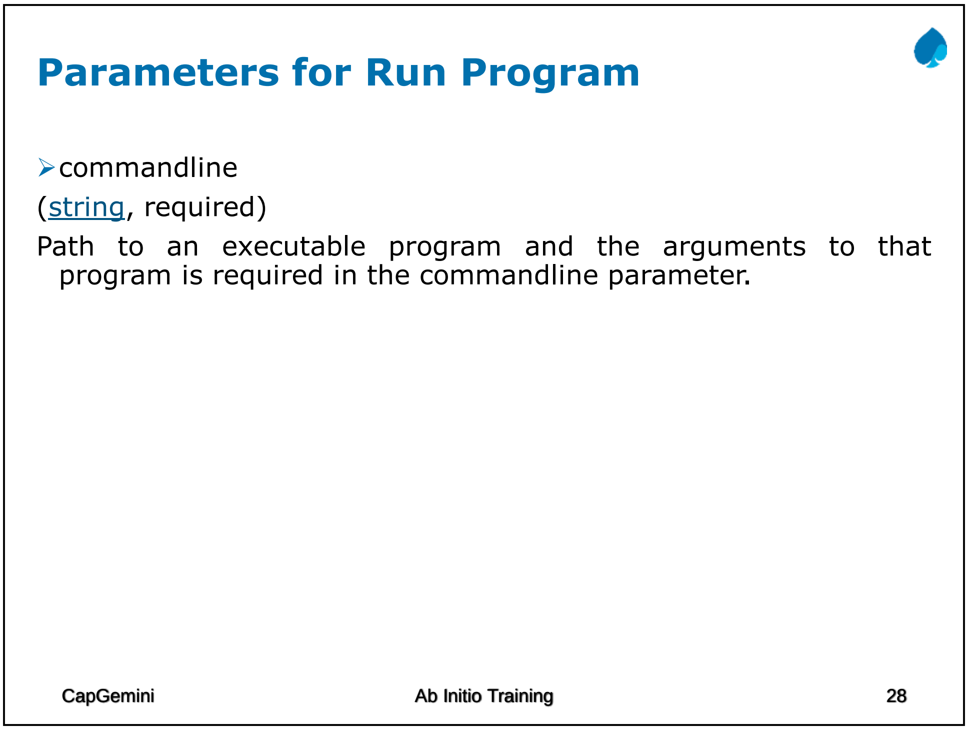
The Gather Logs component:

- Collects log records generated by components through their **log** ports
- Writes a record containing the text from the **StartText** parameter to the file specified in the **LogFile** parameter
- Writes any log records from its **in** port to the file specified in the **LogFile** parameter
- Writes a record containing the text from the **EndText** parameter to the file specified in the **LogFile** parameter

Run Program

- The RUN PROGRAM component is used to run any executable program through a graph.

in
Run Program
out

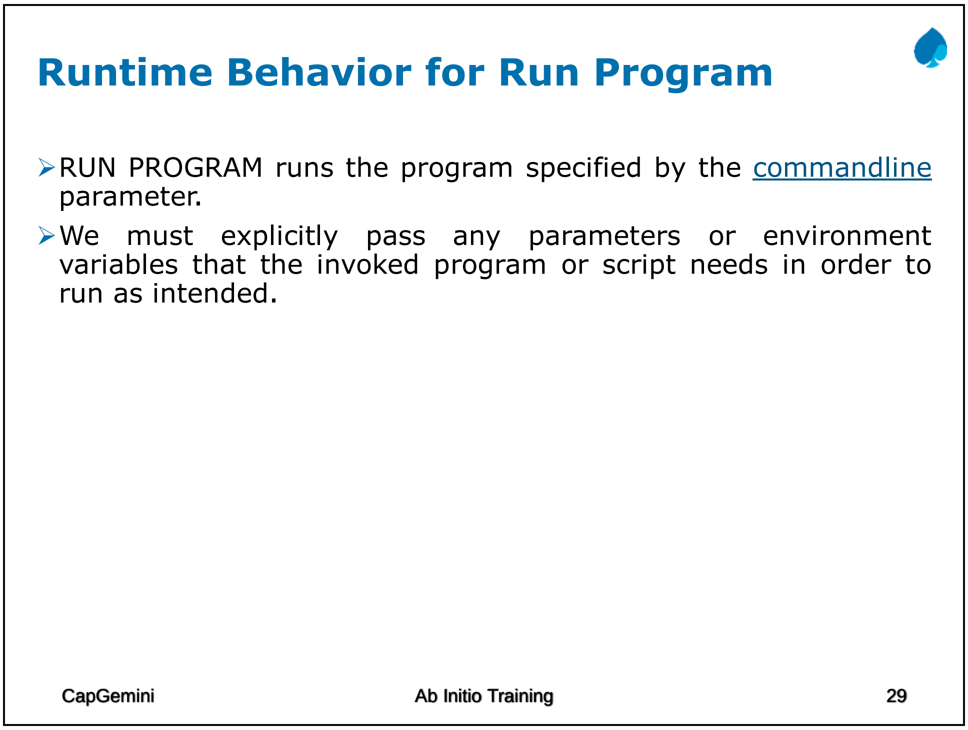


Parameters for Run Program

- commandline

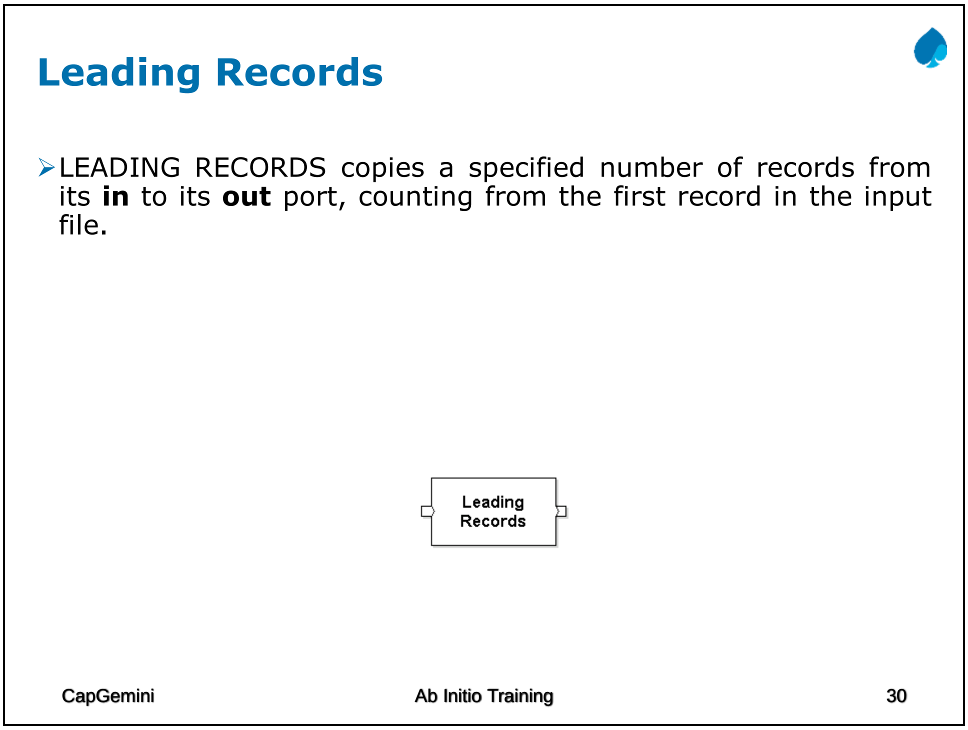
(string, required)

Path to an executable program and the arguments to that program is required in the commandline parameter.



Runtime Behavior for Run Program

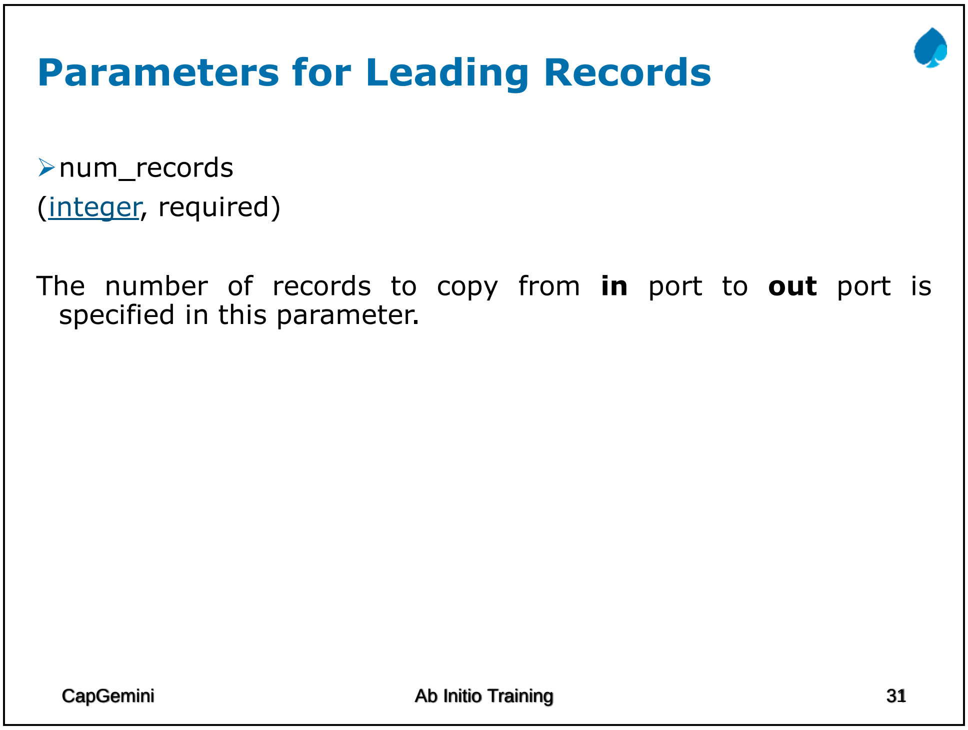
- RUN PROGRAM runs the program specified by the commandline parameter.
- We must explicitly pass any parameters or environment variables that the invoked program or script needs in order to run as intended.



Leading Records

- LEADING RECORDS copies a specified number of records from its **in** to its **out** port, counting from the first record in the input file.

Leading
Records

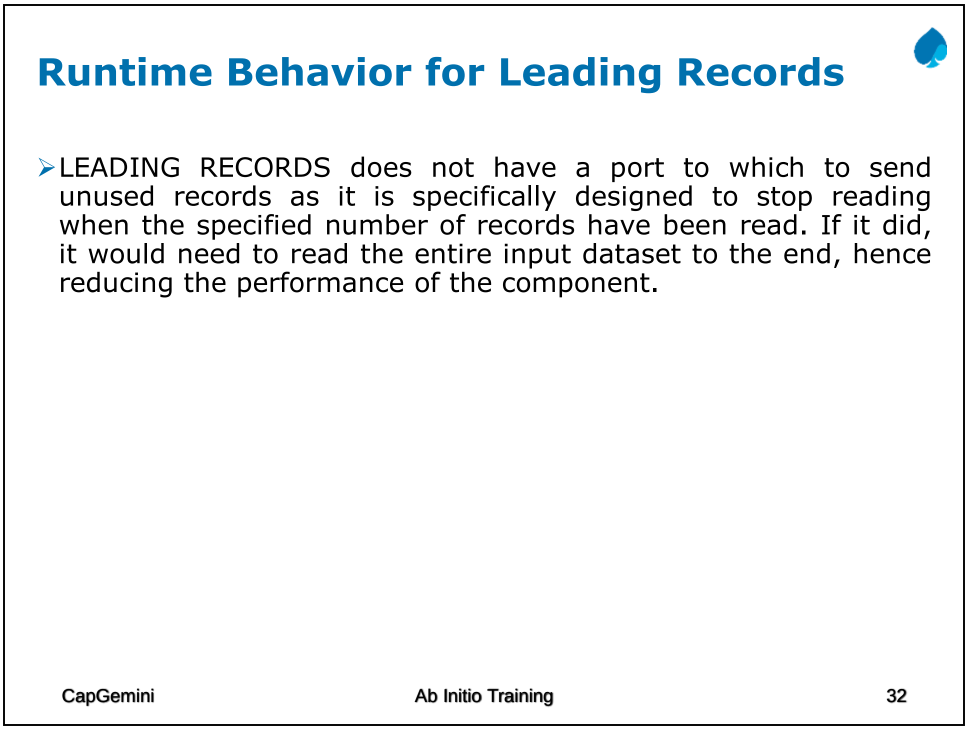


Parameters for Leading Records

- num_records

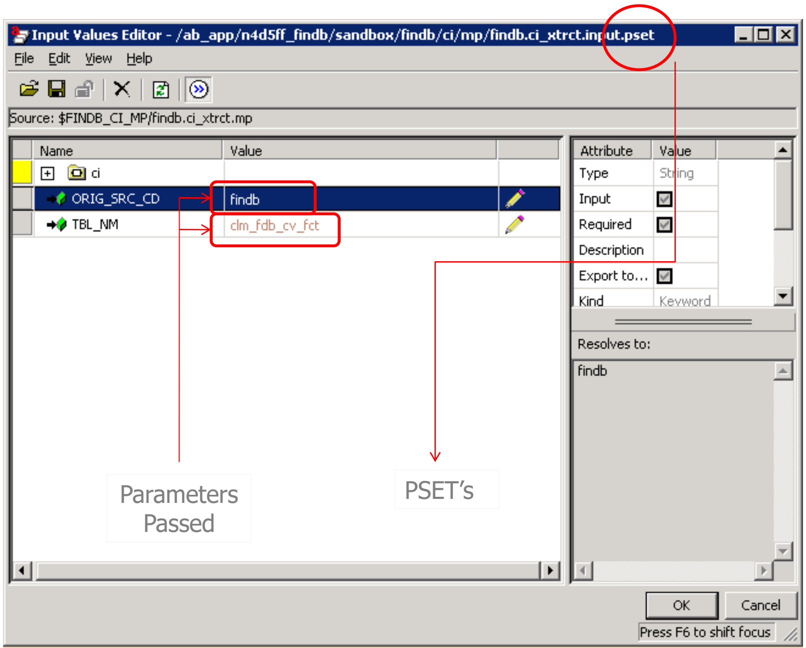
(integer, required)

The number of records to copy from **in** port to **out** port is specified in this parameter.



Runtime Behavior for Leading Records

- LEADING RECORDS does not have a port to which to send unused records as it is specifically designed to stop reading when the specified number of records have been read. If it did, it would need to read the entire input dataset to the end, hence reducing the performance of the component.



Input Values Editor - /ab_app/n4d5ff_findb/sandbox/findb/ci/mp/findb.ci_xtrct.input.pset

Source: $FINDB_CI_MP/findb.ci_xtrct.mp

Name	Value
ci	
ORIG_SRC_CD	findb
TBL_NM	clm_fdb_cv_fct

Attribute	Value
Type	String
Input	☑
Required	☑
Description	
Export to…	☑
Kind	Keyword

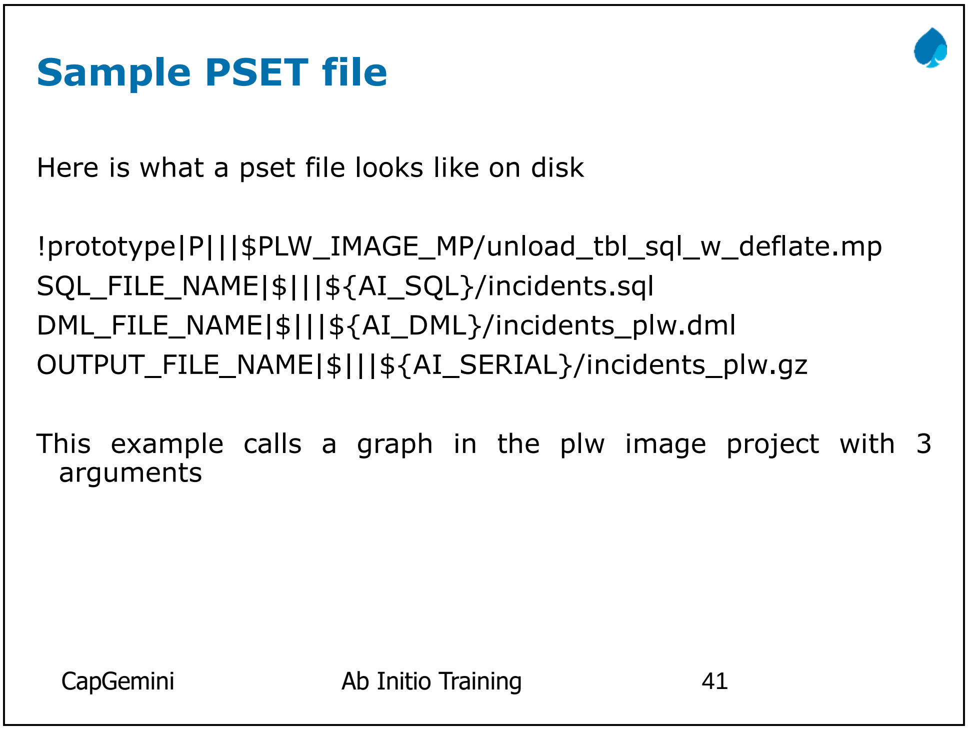
Resolves to:

findb

Press F6 to shift focus

Parameters Passed

PSET's

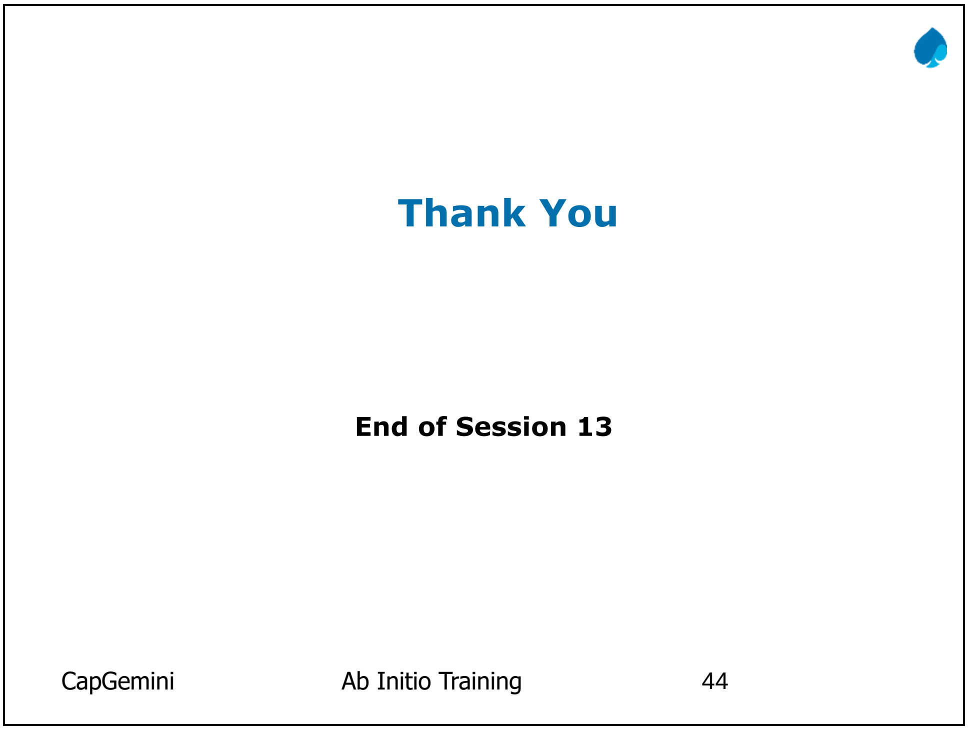


Sample PSET file

Here is what a pset file looks like on disk

```
!prototype|P|||$PLW_IMAGE_MP/unload_tbl_sql_w_deflate.mp
SQL_FILE_NAME|$|||${AI_SQL}/incidents.sql
DML_FILE_NAME|$|||${AI_DML}/incidents_plw.dml
OUTPUT_FILE_NAME|$|||${AI_SERIAL}/incidents_plw.gz
```

This example calls a graph in the plw image project with 3 arguments



Thank You

End of Session 13