# Ab initio Session 4
# Introduction to Ab Initio

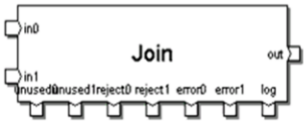
Capgemini

Ab Initio Training 1

## Join

## Join Basic Definition

- Join performs inner, outer, and semi-joins with multiple flows of data records.



# JOIN

- Join reads data from two or more input ports, combines records with matching keys according to the transform you specify, and sends the transformed records to the output port. Additional ports allow you to collect rejected and unused records. There can be as many as 20 input ports.

## Alphabetical List of Join Parameters

**count**
**dedupn**
**driving**
**join-type**
**key**
**limit**
**logging**
**log_input**
**log_intermediate**
**log_output**
**log_reject**
**maintain-order**
**max-core**
**max-memory**
**override- keyn**
**ramp**
**record- requiredn**
**reject-threshold**
**selectn**
**sorted-input**
**transform**

## Parameter Descriptions for Join

**count :** (integer, required)

- An integer *n* from 2 to 20 specifying the total number of inputs (**in** ports) to Join. This in turn determines the number of the following ports and parameters:
  - **unused** *ports*
  - **reject** ports
  - **error** ports
  - **record-required** parameters
  - **dedup** parameters
  - **select** parameters
  - **override-key** parameters Default is **2**.
- Each **in** port (always 2 or more) has a number *n* appended. There can be as many as 20 **in** ports altogether. Each **out***n*, **unused***n*, **reject***n*, and **error***n* port corresponds to an **in***n* port.

## Parameter Descriptions for Join

**sorted-input :** (choice, required)

- When set to **In memory: Input need not be sorted**, Join accepts unsorted input, and permits the use of the **maintain-order** parameter.
- When set to **Inputs must be sorted**, Join requires sorted input, and the **maintain-order** parameter is not available.
- Default is **Inputs must be sorted**.

## Parameter Descriptions for Join

**key :** (key specifier, required)

- Name(s) of the field(s) in the input records that must have matching values for Join to call the transform function.

## Parameter Descriptions for Join

**transform :** (filename or string, required)

- Either the name of the file containing the transform function, or a transform string. In the file specified in the **transform** parameter or in the transform string, create a transform function that has the following characteristics:
  - The transform function takes the number of input arguments specified in the **count** parameter.
  - The first argument is a data record with the record format of the **in0** port. The second argument is a data record with the record format of the **in1** port, and so on.
  - The transform function has an explicit or implicit rule that assigns a value to every field in the output record.

## Parameter Descriptions for Join

**join-type :** (choice, required)

- Choose from the following:
  - **Inner join** — sets the **record-requiredn** parameters for all ports to **True**. **Inner join** is the default. The GDE does not display the **record-required***n* parameters because they all have the same value.
  - **Outer join** — sets the **record-requiredn** parameters for all ports to **False**. The GDE does not display the **record-required***n* parameters because they all have the same value.
  - **Explicit** — allows you to set the **record-requiredn** parameter for each port individually. If you set the **dedupn** parameter to **True** on the driving input, set the **join-type** parameter to **Inner join**. (The driving input is the largest input, as specified by the driving parameter.)

## Parameter Descriptions for Join

**dedup*n* :** (boolean, required)

- Set the **dedup***n* parameter to **True** to remove duplicates from the corresponding **in***n* port before joining. This allows you to choose only one record from a group with matching key values as the argument to the transform function. Default is **False**, which does not remove duplicates.
- If you remove duplicates on this input port before joining it to the driving input, set the **record-required***n* parameter to **True** on all other ports. (The driving input is the largest input, as specified by the driving parameter.)
- There is one **dedupn** parameter associated with each **inn** port.

## Parameter Descriptions for Join

**select*n*** : (expression, optional)

- Filter for records before join function. One per **inn** port; *n* represents the number of an **in** port.

## Parameter Descriptions for Join

**override-key*n* :** (key specifier, optional)

- Alternative name(s) for the key field(s) for a particular **inn** port.
- Supported for Co>Operating System Version:
  - 2.1 and higher with the **sorted-input** parameter set to **In memory: Input need not be sorted**
  - 2.2.2 and higher with the **sorted-input** parameter set to **Inputs must be sorted** There is one **override-key*n*** parameter per **inn** port. The *n* corresponds to the number of an **in** port.
- To use key field(s) other than the key field(s) specified in the **key** parameter for a particular **inn** port, specify the key field(s) you want to use in the corresponding **override-key*n*** parameter. Default is **0.0**.

## Parameter Descriptions for Join

**max-memory** : (integer, required)

- Maximum memory usage in bytes before Join writes temporary files to disk. Only available when the **sorted-input** parameter is set to **Inputs must be sorted**.
- The default value is **8388608** bytes (8 megabytes). Start by using this default, and then adjust to higher or lower values as necessary if you encounter performance difficulties. It is very unlikely you will ever need to change the value of this parameter.

## Parameter Descriptions for Join

**driving** : (integer, required)

- Number of the port to which you connect the driving input. The driving input is the largest input. All other inputs are read into memory.
- The **driving** parameter is only available when the **sorted-input** parameter is set to **In memory: Input need not be sorted**. For example, suppose the largest input to be joined is on the **in1** port. Specify a port number of **1** as the value of the **driving** parameter. The Join component reads all other inputs to the join, for example, **in0**, and **in2**, into memory.
- Default is **0**, which specifies that the driving input is on port **in0**.

## Parameter Descriptions for Join

**maintain-order :** (boolean, required)

- Set to **True** to ensure that records remain in the original order of the driving input. (The driving input is the largest input, as specified by the **driving** parameter.) Default is **False**.
- Only available when the **sorted-input parameter** is set to **In memory: Input need not be sorted**. If the **sorted-input** parameter is set to **Inputs must be sorted**, and all inputs are sorted on the fields given in the **key** parameter, then the output maintains the sort order on that key without the use of this parameter.

## Parameter Descriptions for Join

**maintain-order :** (boolean, required)

- If any inputs, other than the driving input, are too large to fit within the memory limit specified by **max-core**, and you set **maintain-order** to
- **False** — Join stores some of its intermediate results in temporary files on disk. This alters the order of records in the driving input.
- **True** — Join stops execution of the graph. Even if you leave **maintain-order** set to **False**, Join groups together all output records for a given driving input record. If the driving input is grouped by key value, you can still use components downstream that require records grouped by key value.

## Parameter Descriptions for Join

**max-core :** (integer, required)

- Maximum memory usage in bytes. Only available when the **sorted-input** parameter is set to **In memory: Input need not be sorted**. The default value is **67108864** bytes (64 megabytes).
- If the total size of the intermediate results Join holds in memory exceeds the number of bytes specified in the **max-core** parameter, Join writes temporary files to disk

## Runtime Behavior of Join

- The Join component:
- Reads data records from multiple **inn** ports.
- Applies the expression in any defined **select***n* parameter to the records on the corresponding **in***n* port.
  - If the expression evaluates to **0** for a record, Join does not process the record, and it does not appear on any output port.
  - If the expression produces NULL for a particular record, Join writes a descriptive error message and stops graph execution.
  - If the expression evaluates to anything other than **0** or NULL for a particular record, Join processes the record.

## Runtime Behavior of Join

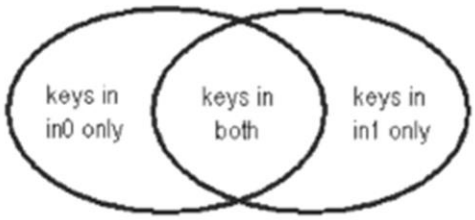
- If you do not supply an expression for a **select***n* parameter, Join processes all the records on the corresponding **in***n* port.
- Operates on the records that have matching key values using a multi-input transform function.
- Writes the result to the **out** port. If you connect a flow to an **unused***n* port, Join writes to the **unused***n* port, from the corresponding **in***n* port, any of the selected records that it does not pass through the transform function. In other words, Join writes the following records to **unusedn** ports:
- For an inner join — all unmatched records
- For an outer join — no records, since Join passes all records through the transform function
- For an explicit join — records for which the transform is not called
- For an input port with the **dedup***n*_parameter set to **True** — records with duplicate key values

## Runtime Behavior of Join

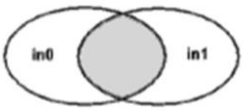
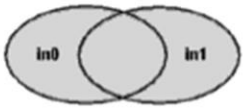
- Thus, the set of records that Join passes through the transform function is mutually exclusive with the set of records that come out the **unusedn** port, and the two sets are also collectively exhaustive. The result is that all selected records are accounted for exactly once.
- If the transform function returns NULL, Join writes:
- Each input record to the corresponding **rejectn** port.

## Join Types

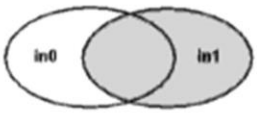
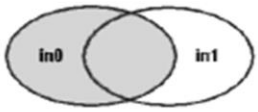
- Inner Join (Equi-Join)
- Full Outer Join (Cartesian Product)
- Explicit Join (Left or Right Outer Join



## Join Types : Inner & Outer

| | |
|---|---|
| **Case 1**<br>**Inner Join join-type** |  |
| **Case 2**<br>**Full Outer Join join-type** |  |

## Join Types : Explicit

| | |
|---|---|
| **Case 3a**<br>**Explicit join-type:**<br>**record-required0: False**<br>**record-required1: True** |  |
| **Case 3b**<br>**Explicit join-type:**<br>**record-required0: True**<br>**record-required1: False** |  |

# Joining of Data

```
record
  decimal(4) id;
  string(6) name;
  string(8) city;
  decimal(3) amount;
end
```

Record Format In0

Number of Visits

| name | city | Visits |
|---|---|---|
| James | London | 2 |
| Black | New York | 1 |
| Jim | Bristol | 2 |
| Tom | Boston | 3 |
| Mick | Chicago | 4 |

Input Data1

Cost of Visits

| id | dt | cost |
|---|---|---|
| 665 | 20050621 | 633.21 |
| 322 | 20050306 | 342.12 |
| 345 | 20050812 | 662.54 |
| 212 | 20050425 | 12.26 |

Input data 2

```
record
  decimal(4) id;
  date("YYMMDD") dt;
  decimal(9.2) cost;
end
```

Record Format In1

```
record
  decimal(4) id;
  string(8) city;
  decimal(3) amount;
  date("YYYY/MM/DD")dt;
end
```

Output

## Input data sorted before Join

| id | name | city | amount |
|---|---|---|---|
| 121 | Black | New York | 32 |
| 212 | Jim | Bristol | 66 |
| 322 | Tom | Boston | 15 |
| 345 | James | London | 2 |
| 492 | Mick | Chicago | 25 |

Input 0 Data is Sorted by id before Join

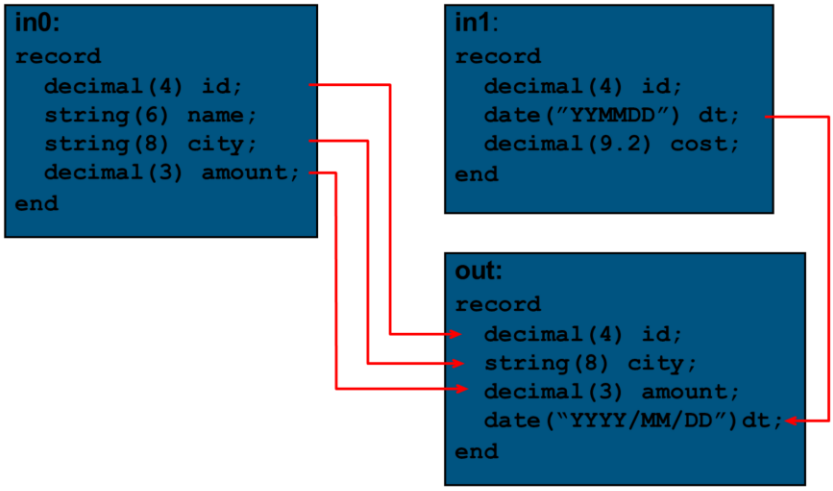
| id | dt | cost |
|---|---|---|
| 121 | 20050425 | 12.26 |
| 322 | 20050306 | 342.12 |
| 345 | 20050812 | 662.54 |
| 665 | 20050621 | 633.21 |

Input 1 Data is sorted by id before Join

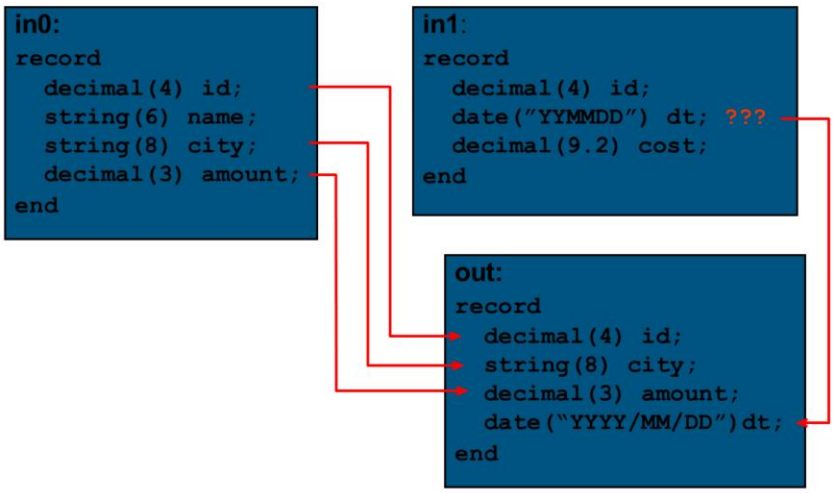
## The Join Component

- Join performs a join of inputs. By default, the inputs to join must be sorted and an inner join is computed.
- Note: The following slides and the on-line example assume the join-type parameter is set to 'Outer', and thus compute an outer join.

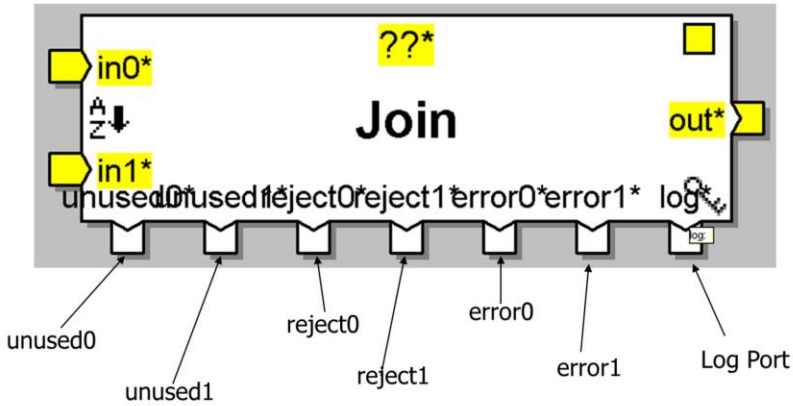
## Building the Output Record



## What if the in1 record is missing?



## Ports of Join Component



## Join Type : Scenario

**Customer**

| Cust Id | Cust Name |
|---|---|
| C1 | Name1 |
| C2 | Name2 |
| C3 | Name3 |

**Transaction**

| Tran Id | Cust Id | Tran Amt |
|---|---|---|
| T1 | C2 | 34 |
| T2 | C1 | 45 |
| T3 | C1 | 67 |
| T4 | C2 | 23 |
| T5 | C4 | 33 |

**Key**

Fields in Output of Join

| Cust Id | Cust Name | Tran Amt |
|---|---|---|

How many records will be generated at the output of Join

# Join Type : Scenario

**Customer**

| Cust Id | Cust Name |
|---|---|
| C1 | Name1 |
| C2 | Name2 |
| C3 | Name3 |

**Key**

**Transaction**

| Tran Id | Cust Id | Tran Amt |
|---|---|---|
| T1 | C2 | 34 |
| T2 | C1 | 45 |
| T3 | C1 | 67 |
| T4 | C2 | 23 |
| T5 | | 33 |

**Fields in Output of Join**

| Cust Id | Cust Name | Tran Amt |
|---|---|---|

How many records will be generated at the output of Join

# Join Type : Scenario

**Customer**

| Cust Id | Cust Name |
|---|---|
| C1 | Name1 |
| C2 | Name2 |
| | Name3 |

**Transaction**

| Tran Id | Cust Id | Tran Amt |
|---|---|---|
| T1 | C2 | 34 |
| T2 | C1 | 45 |
| T3 | C1 | 67 |
| T4 | C2 | 23 |
| T5 | C4 | 33 |

**Key**

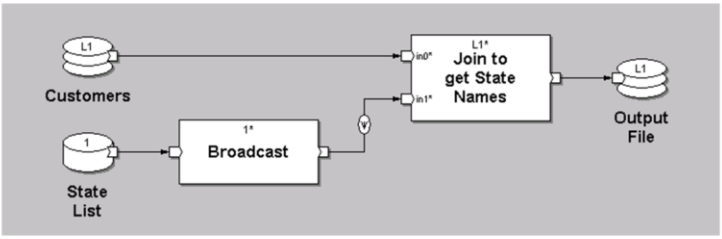
**Fields in Output of Join**

| Cust Id | Cust Name | Tran Amt |
|---|---|---|

How many records will be generated at the output of Join

# Minimize operations on large data

- When joining a very small dataset to a very large dataset, it may be more efficient to broadcast the small dataset or use it as a lookup file rather than repartition and re-sort the large dataset.



- Minimize sorted join component and if possible replace them by in-memory join/hash join.
- If the two inputs are huge then use sorted join, otherwise use hash join with proper driving port.

# Exercise 3

# Thank You

**End of Session 4**