ABAP Part I

Lesson 06: Internal Tables - I

Lesson Objectives

After completing this lesson, participants will be able to know:

- To Define an Internal Table and understand its attributes
- Types of Internal Tables
- To Add, Read, Update and Delete Data from an internal Table
- To Sort the Contents of an Internal Table

Internal Tables

Internal table provides a means of taking data from a fixed structure and storing it in working memory in ABAP

An internal table is a data object that contain any rows with any data type.

Internal Tables fulfill the function of Arrays

A very important use of internal tables is for storing and formatting data from a database table within a program.

Work Area In Internal tables

Work areas are single rows of data.

They should have the same format as any of the internal tables.

It is used to process the data in an internal table one line at a time.

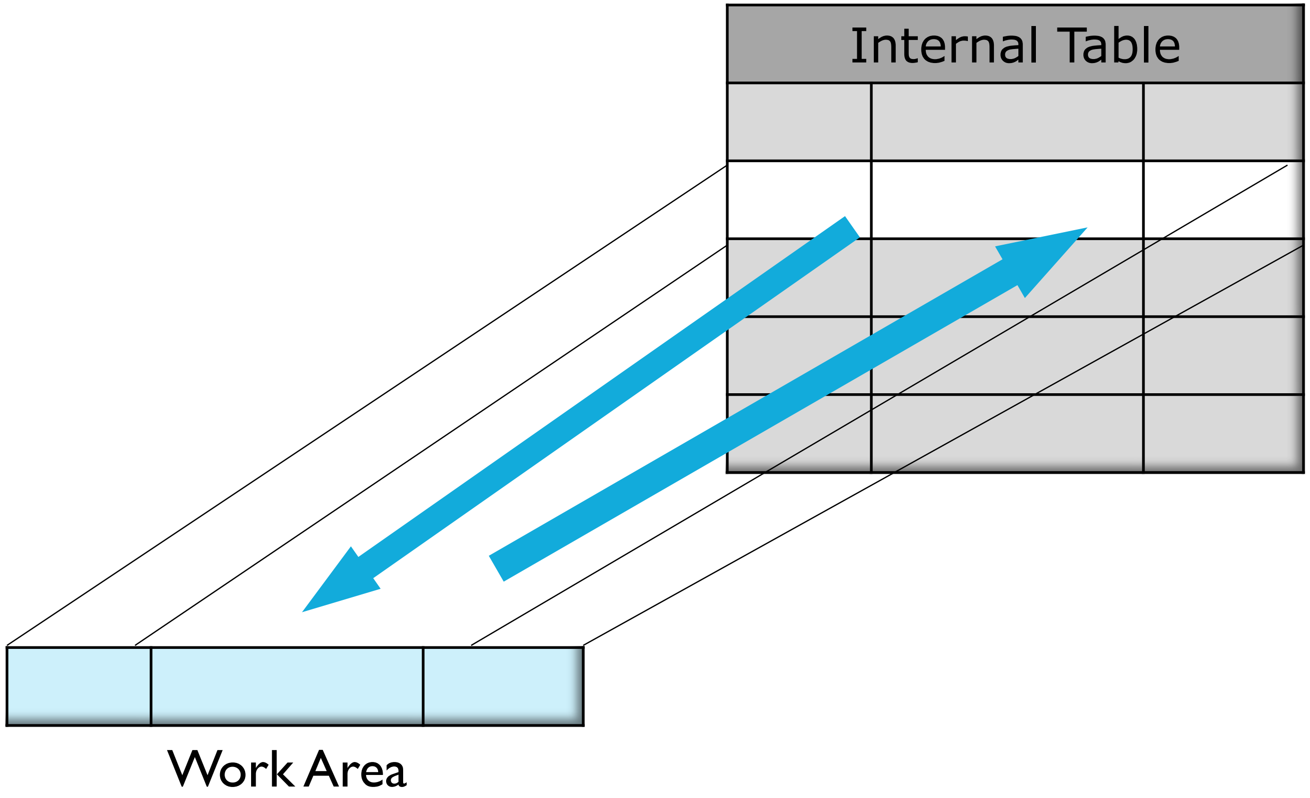
Access Methods to Individual Table Entries

There are two ways to access the individual rows of an Internal Table

- Accessing the Internal Table Rows Using a Work Area
  - The data in the table is not directly accessed but through another Data Object referred as a Work Area
  - Work Area must be compatible with the line type of internal table
  - When a data is read from the table, the data overwrites the current contents of the Work Area
  - When data is written to the Internal Table, it must be placed in the Work Area and then transferred to the Internal Table

- If the internal table has a Header Line, the Header Line can act as a Work Area.

Access Using a Work Area



Internal Tables with Header Line and Without Header Line

Internal tables can be created:

- Without Header Lines
- With Header Lines

When using internal tables with header lines, the header line and the body of the table have the same name.

If there is an internal table with header line, to address the body of the table, place brackets after the table name (<itab>[]).

If not, ABAP interprets the name as the name of the header line and not of the body of the table.

It is possible to avoid this potential confusion by using internal tables without header lines.

Internal Tables without Header Line

- Work area is to be explicitly specified when we need to access such tables.
- Hence these tables cannot be accessed directly.

Internal Tables without Header Line - Example

```
REPORT Z.
DATA ITAB TYPE TABLE OF I. "without header line
DATA WA TYPE I.  "work area
WA = 10.
APPEND WA TO ITAB.
WA = 20.
APPEND WA TO ITAB.
WA = 30.
APPEND WA TO ITAB.
WA = 40.
APPEND WA TO ITAB.
WA = 50.
APPEND WA TO ITAB.

LOOP AT ITAB INTO WA.
  WRITE:/ WA.
ENDLOOP.
```

Demo

Program on Internal table without header line

Internal Tables with Header Line

- Here the system automatically creates the work area.
- When using internal tables with header lines, the header line and the body of the table have the same name.
- The work area has the same data type as internal table.
- This work area is called the HEADER line.
- It is here that all the changes or any of the action on the contents of the table are done.
- As a result of this, records can be directly inserted into the table or accessed from the internal table directly

Internal Tables with Header Line - Example

```
REPORT Z.
ITAB = 10. "Header line
APPEND ITAB TO ITAB. "APPEND HEADERLINE TO INTERNAL TABLE
ITAB = 20.
APPEND ITAB. "Implies APPEND ITAB TO ITAB
ITAB = 30.
APPEND ITAB.
ITAB = 40.
APPEND ITAB.
ITAB = 50.
APPEND ITAB.

LOOP AT ITAB. "IMPLIES LOOP AT ITAB INTO ITAB
  WRITE ITAB.
ENDLOOP.
```

Demo

Program on Internal table with header line

Internal Tables Data Type

The data type of an Internal table is

- A table type defined in ABAP Dictionary OR Using Types or Data is specified fully using:
  - Row Type
  - Table Category
  - Table key

Internal Tables Data Type

Row Type: The row type of an internal table can be any data type. If the row type is structured, the individual components of a row are also known as the columns of the internal table.

Table Category: The table category defines how an internal table is administered and how its individual rows can be accessed. There are three table categories:

- Standard
- Sorted
- Hashed

Standard Table

Have an internal linear index

Records are accessed by index or keys

The response time for key access is proportional to number of entries

The key is always non-unique

Sorted Table

Always saved sorted by the key

Have an internal index

Records can be accessed by table index or key

Uses Binary Search for access

Key can be Unique or non-Unique

Hashed Table

Has no linear index

Only accessed using its key

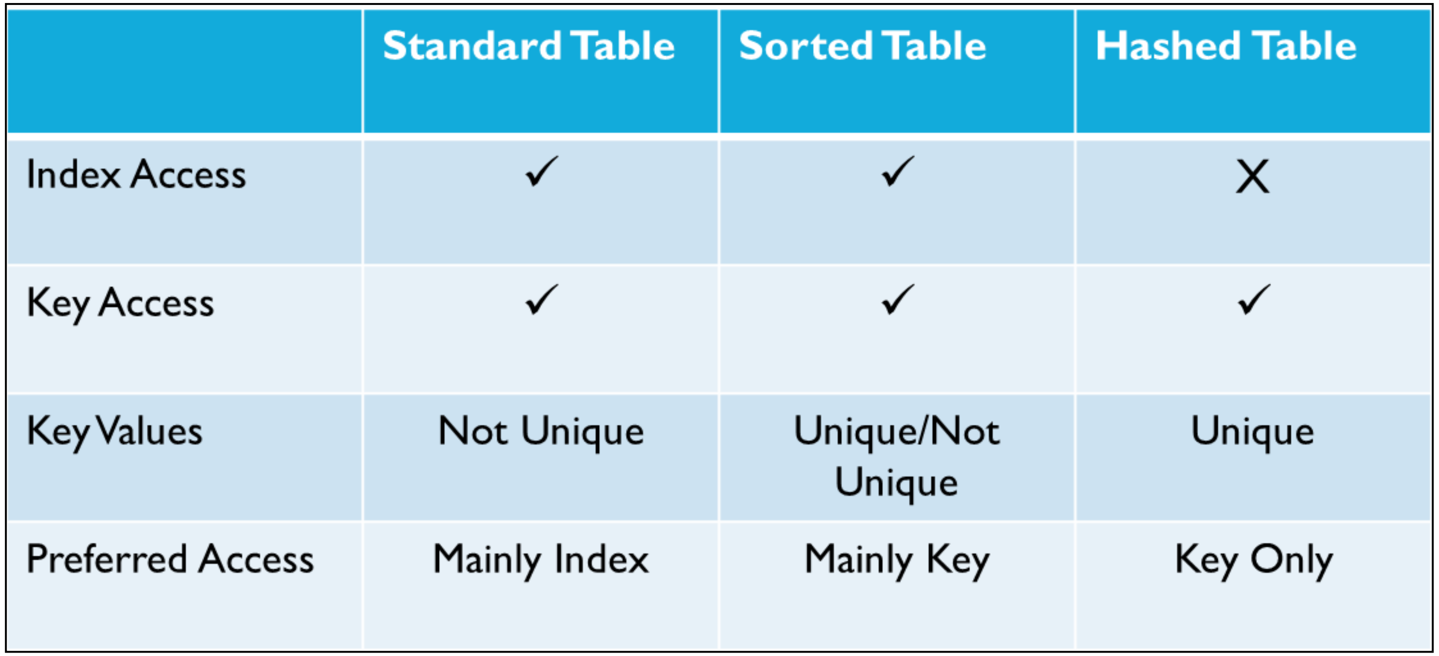
Key must be Unique

Uses Hash Algorithm for accessing records

Response time is constant

Operations on Individual Lines

Key and Index Access



	Standard Table	Sorted Table	Hashed Table
Index Access	✓	✓	X
Key Access	✓	✓	✓
Key Values	Not Unique	Unique/Not Unique	Unique
Preferred Access	Mainly Index	Mainly Key	Key Only

Table key

There are two possible key types for internal tables

- Primary keys

And

- Optional secondary keys.

Every internal table has a primary key, which is either

- standard key
  - Standard key is the primary table key of an internal table, whose key fields are all table fields with character-like data types and byte-like data types.

or

- self-defined key

Table key

Definition of the table keys in an internal table.

The following can be defined:

- A primary table key using key.
  - Every internal table has a primary key.
  - The primary key must be defined explicitly for sorted and hashed tables.
  - The primary key specification can only be omitted for standard tables.
  - The primary table key is then automatically defined as a non-unique standard key.

- Several secondary table keys
  - Specification of secondary keys is optional for all table types.
  - If no secondary key is specified, the internal table has only one primary key.

Primary Table Key

The components of the primary table key are declared using the UNIQUE|NON-UNIQUE KEY additions of the statements TYPES, DATA.

E.g.

```
TYPES sbook_tab
      TYPE STANDARD TABLE
      OF sbook
      WITH NON-UNIQUE KEY carrid.
```

Reference to Declared Internal Table Types

Declare internal table objects using the LIKE or TYPE.

```
DATA <itab> TYPE <type>|LIKE <obj> [WITH HEADER LINE].
```

- Here, the LIKE addition refers to an existing table object in the same program.

The TYPE addition can refer to an internal type in the program declared using the TYPES statement, or a table type in the ABAP Dictionary.

The optional addition WITH HEADER LINE declares an extra data object with the same name and line type as the internal table.

Defining Internal Tables with Global Types

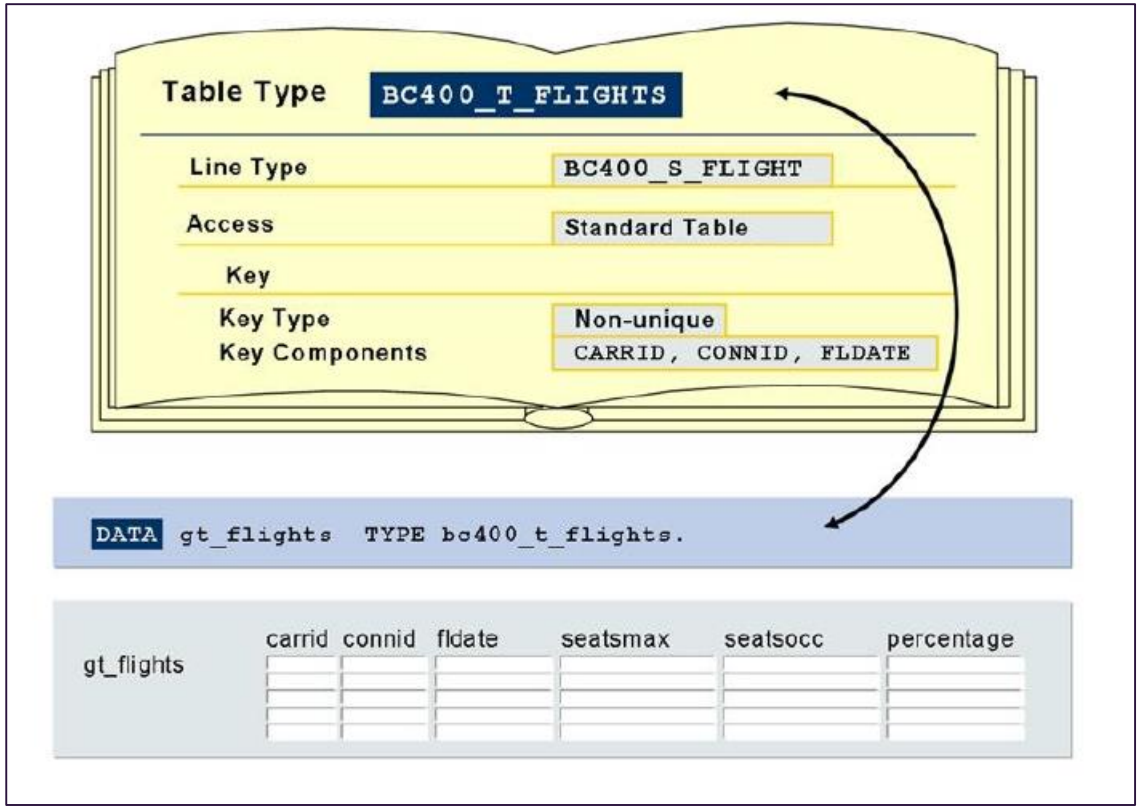


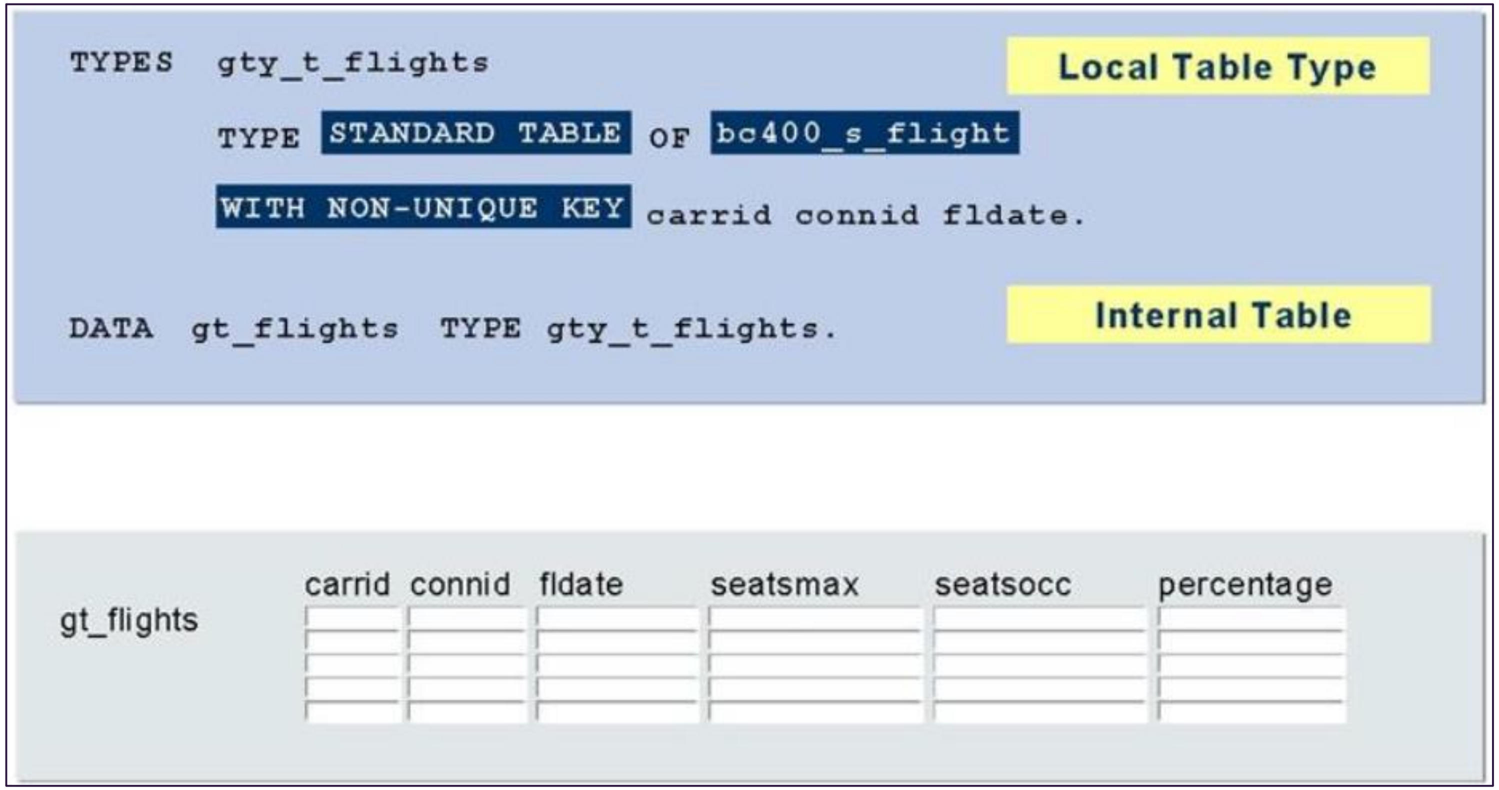
Table Type BC400_T_FLIGHTS

Line Type	BC400_S_FLIGHT
Access	Standard Table
Key	
Key Type	Non-unique
Key Components	CARRID, CONNID, FLDATE

```
DATA gt_flights  TYPE bc400_t_flights.
```

	carrid	connid	fldate	seatsmax	seatsocc	percentage
gt_flights						

Defining Internal Tables with Local Types



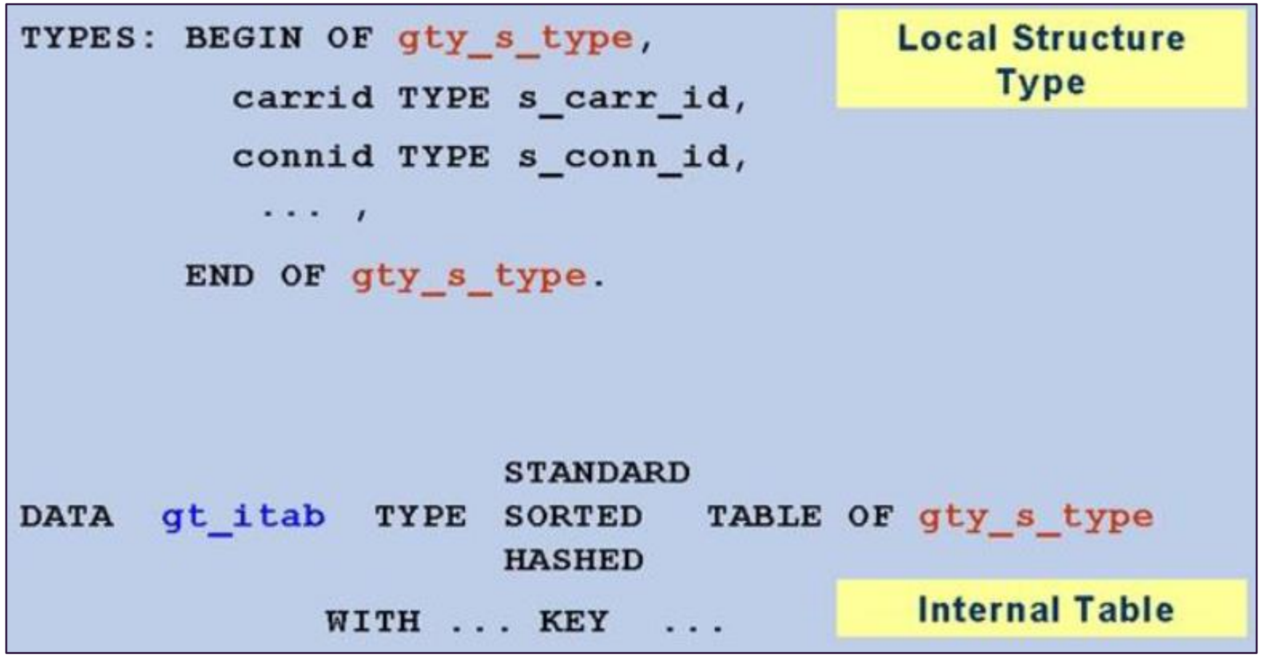
```
TYPES  gty_t_flights
       TYPE STANDARD TABLE OF bc400_s_flight
       WITH NON-UNIQUE KEY carrid connid fldate.

DATA  gt_flights  TYPE gty_t_flights.
```

Local Table Type

Internal Table

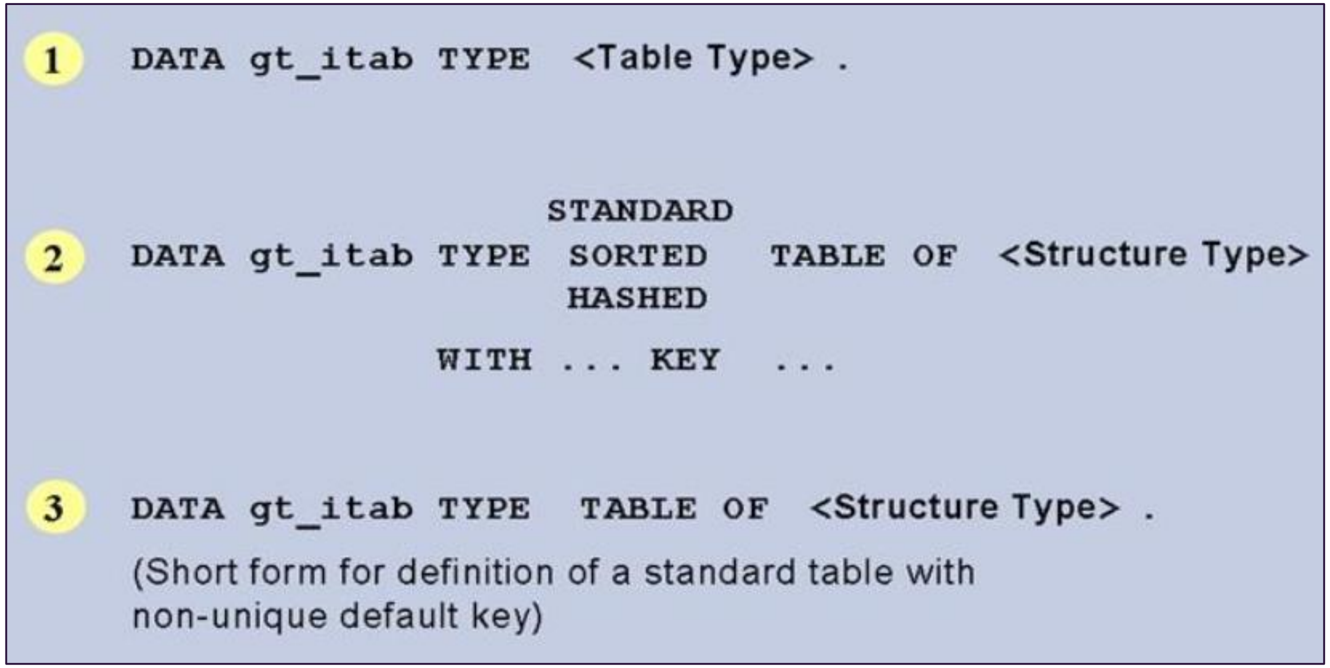
	carrid	connid	fldate	seatsmax	seatsocc	percentage
gt_flights						

Independent Definition of Internal Tables



```
TYPES: BEGIN OF gty_s_type,
           carrid TYPE s_carr_id,
           connid TYPE s_conn_id,
            ... ,
       END OF gty_s_type.


                        STANDARD
DATA   gt_itab   TYPE   SORTED    TABLE OF gty_s_type
                        HASHED
                WITH ... KEY   ...
```

Possible Definition of Internal Tables



```
1  DATA gt_itab TYPE  <Table Type> .


                        STANDARD
2  DATA gt_itab TYPE  SORTED    TABLE OF  <Structure Type>
                      HASHED
                   WITH ... KEY  ...


3  DATA gt_itab TYPE  TABLE OF  <Structure Type> .
```

(Short form for definition of a standard table with
non-unique default key)

Processing Statements for Internal Tables

The following can be used to Read records from an Internal table:

- Loop AT
- Read Table

Processing Statements for Internal Tables: Loop AT

The Syntax is:

```
 LOOP AT itab result [cond].
    ...
  ENDLOOP.
```

Where Result is

```
... { INTO wa }
```

Cond is:

```
... [FROM idx1] [TO idx2]
    [WHERE log_exp|(cond_syntax)] ..
```

Demo

Read an Internal table using Loop at Statement

Processing Statements for Internal Tables: Read Table

The Syntax is:

```
READ TABLE itab { table_key
                | free_key
                | index } result.
```

Processing Statements for Internal Tables: Read Table - Index

READ TABL itab Index idx

- If the addition **INDEX** is used, the statement **READ** reads the row of the row number specified in **idx**

Demo

Read an Internal table using Index

Processing Statements for Internal Tables: Read Table – free key

READ TABLE - free_key

Effect

Specifies a free search key.

Syntax

.. WITH KEY *{* comp1 = operand1 comp2 = operand2 ... *[*BINARY SEARCH*]* *}*

Demo

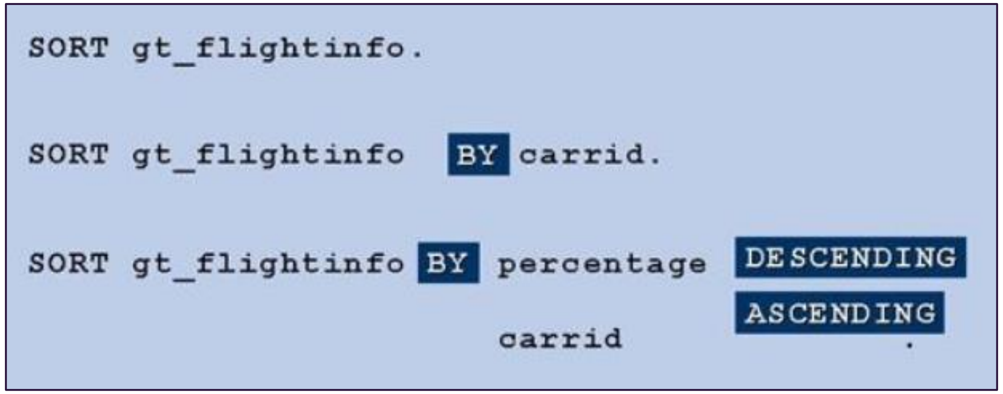
Read an Internal table using key

Internal Tables - Processing

Sorting Internal Tables

- Syntax

SORT itab [ASCENDING|DESCENDING]



```
SORT gt_flightinfo.

SORT gt_flightinfo  BY carrid.

SORT gt_flightinfo BY percentage  DESCENDING
                      carrid      ASCENDING
                                           .
```

Demo

Program on Insert a record and Insert Multiple records

Appending Lines

Appending a Single Line

- APPEND line TO itab.

Appending Multiple Lines

- APPEND LINES of itab1 TO itab2.

Processing Table Entries in Loops

Syntax:

```
LOOP AT itab result condition.
     Statements…
ENDLOOP.
```

- Depending on the table types the lines are processed

- Standard and Sorted Table
  - Processed based on Index
  - The system field SY-TABIX stores the index of the current line

- Hashed Table
  - If the table is not sorted, the lines are accessed in the inserted order
  - SY-TABIX = 0 , within the processing block.

Changing Lines

Changing Lines of an Internal Table

```
MODIFY  itab FROM wa [TRANSPORTING f1 f2 ...].
```

Changing Internal Table Contents based on Condition

```
MODIFY itab FROM wa TRANSPORTING f1 f2 ... WHERE cond.
```

- Changing the contents of an Internal Table Using an Index

```
MODIFY itab FROM wa [INDEX idx] [TRANSPORTING f1 f2... ].
```

Deleting Lines

Deletion Using Table Key

```
DELETE TABLE itab WITH TABLE KEY k1 = f1 … kn = fn.
```

Delete the Internal Table Contents Based on a Condition

```
DELETE itab WHERE cond.
```

Deleting Duplicate Records

```
 DELETE ADJACENT DUPLICATE ENTRIES FROM itab [COMPARING f1
f2 … |ALL FIELDS].
```

Deleting Lines – contd..

Deleting a line specifying the Index

```
DELETE itab [INDEX idx].
```

Deleting Several Lines

```
DELETE [FROM n1] [TO n2] [WHERE condition].
```

Demo

Program on deleting from Internal tables

Obsolete Statements

Using HEADER LINE as Work Area.

Example:

```
 TYPES: BEGIN OF line,
   num TYPE i,
   sqr TYPE i,
END OF line.
DATA it_tab TYPE  TABLE OF line WITH UNIQUE KEY col1 WITH HEADER LINE.
DO 5 TIMES.
   It_tab-num = sy-index.
   It_tab-sqr = sy-index ** 2.
   INSERT TABLE it_tab.
  ENDDO.
It_tab-sqr = 100.
  MODIFY TABLE it_tab.
It_tab-num = 4.
  DELETE TABLE it_tab.
```

Summary

In this lesson, you have learnt:

- To Define an Internal Table and understand its attributes
- Types of Internal Tables
- To Add, Read, Update and Delete Data from an internal Table
- To Sort the Contents of an Internal Table

Review Question

Question 1: ______________ is a field string with the same structure as a row of the body, but it can hold a single row.

Question 2: ________ adds a single row to an internal