Internal Tables:

* Internal table is a temporary table that contains the records of an ABAP program while it is being executed.
* It exists only during the execution of the program. So the records are discarded when the program terminates.
* It is declared in an ABAP program when u need to retrieve the data from database tables.
* It stores data in the form of rows and columns.
* **Individual rows are accessed by index or a key.**

WorkArea:

* By default no operation can be performed on internal table so we need workarea.
* Internal table is accessed by the concept of WorkArea.
* Work area is a temporary memory space that helps in reading and modifying the data of an internal table **line by line.**
* Work area must have the same structure as that of the underlying internal table.

Syntax of internal table and workarea: (3.EDITInternal Table\_Demo3)

DATA : <itab\_name> TYPE TABLE of <DBtable name>,

<wa\_name> TYPE <DBtable name>.

OR

DATA: <itab\_name> TYPE TABLE of <DBtable name> OCCURS 0 WITH HEADER LINE. -no work area will be used here so no need to create 1 exclusively. Occurs 0 means initial size of itab is 8 kb and then it increases.

<wa\_name> TYPE <DBtable name>.

OR

DATA : BEGIN OF itab\_name OCCURS 0,

Var1 type I,

var2 type I

END OF itab\_name.

Wa TYPE itab\_name.

OR

TYPES : BEGIN OF struct\_name,

Var1 type I,

var2 type I

END OF struct\_name.

DATA: itab\_name TYPE STANDARD TABLE OF struct\_name.

DATA : wa TYPE struct\_name.

OR

wa LIKE LINE OF itab\_name.

**Types of internal table:**

|  |  |  |
| --- | --- | --- |
| **Standard** | **Sorted** | **Hashed** |
| By default created table. | Records are automatically sorted whenever new records are added. It does not allow entry of unsorted records. | Used when there is large amount of data. |
| Uses key operation or index operation to read data | Uses key operation or index operation to read data | Use only Key operation |
| Linear or binary search to read records | Only Binary search | Hash algorithm to search . |
| Append can b used to append wa to itab.  insert wa INTO TABLE itab also works. | Append can b used to append wa to itab.  insert wa INTO TABLE itab also works.  Read Table itab into wa1 index 2. | Only insert wa INTO TABLE itab works. |

To read from internal table use :

 READ TABLE stud1 into wa1 index 2.

 WRITE : / sy-tabix , wa1-ROLLNO , wa1-NAME.

OR

**Select** is only for DB tables

**Read** is for internal tables

SY-INDEX is used to describe the number of iteration with in the DO..ENDDO, WHILE ...ENDWHILE....

SY-TABIX is used to define the iteration with in the internal table like between LOOP AT & ENDLOOP.

**Operations on Internal Table:**

1. **Append :**  Used to add 1 single record from workarea to internal table. Record is added at the bottom.

**APPEND wa INTO itab.**

***Loop syntax:***

**LOOP AT i\_table INTO wa.**

1. **INSERT :** Used to insert record from a workarea to internal table at a specified location.

**INSERT wa INTO itab INDEX indexno.**

insert wa1 into table stud1. (CANNOT WRITE INDEX NO here)

1. **CLEAR:** Used to delete the data from workarea(WA).

**CLEAR** wa\_name.

1. **REFRESH :** Used to delete data from internal table.

**REFRESH** itab\_name.

1. **FREE :** Used to delete data from internal and workarea.

**FREE** internal\_tablename/wa.

CLEAR is used to clear a value in a work area or in a variable.

REFRESH is used to clear all values in an internal table.

FREE is used to clear (free) memory of an internal table or work area. We all know whenever we declare an internal table or work area, 8kb memory will be allocated.

1. **READ : 3 Ways of reading from workarea- loop , index and key(Hash tables can be read only by key and loop).**

READ TABLE stud1 into wa1 index 2.

1. **DESCRIBE :** Used to find the total no of records in an internal table.

**DESCRIBE table itab.**

**sy-tfill --- no of rows and value of Iines (i)**

**,sy-tleng – length of the rows in bytes depending on the data type of the variable structure**

**and sy-toccu.--** as filled with the value of the initial main memory requirements for the addressed internal table. Also given to initial size value(j).

1. **SORTED BY:**  Has to be used with initial size ONLY. Initial size has to be greater than 0. The output will display only those many no of sorted append statements as mentioned in the initial size. The non sorted values will get appended after the size is completed.

**Example scenarios :**

**If Initial size is** 1, sorted append statements are 3 and unsorted append statements are 2 – then it will print total 3 statements – which are-> 1st sorted value(in descending order) and the 2 unsorted values.

**If Initial size is** 4, sorted append statements are 3 and unsorted append statements are 2 – then it will print total 5 statements – which are-> 3 sorted values(in descending order) and the 2 unsorted values.

1. **COLLECT:** Used for inserting components of workarea into internal table by avoiding duplicates and also in summarized way.
   * **First it will check in internal table for any record matching with the key in work area data.**
   * COLLECT is used to summate numerical entries in an internal table by looking all other non-numeric values. If all non-numeric values are same it sums up numeric values and updates proper record, but if any non-numeric field is not same it appends a new record.
   * Collect statement calculates the sub total of non-character fields(P,I,F) by taking character fields as key. In case all fields are of only character type then it will work similar to the append statement.
2. **SUM.**

The statement SUM can only be specified within a loop starting with [LOOP](SAPEVENT:ABAPLOOP_AT_ITAB), and is only considered within a [AT-ENDAT](SAPEVENT:ABAPAT_ITAB) control structure. Prerequisites for using the statement SUM include using the addition INTO in the LOOP statement, and that the specified work area wa is [compatible](SAPEVENT:ABENCOMPATIBLE_GLOSRY) with the row type of the internal table. In addition, SUM cannot be used when the row type of the internal table itab contains components that are tables.

The statement SUM calculates the sums of the components with [numerical data type](SAPEVENT:ABENNUMERIC_DATA_TYPE_GLOSRY) of all rows in the current group level and assigns the sums to the components of the work area wa. In the control levels FIRST, LAST, and outside of an AT-ENDAT control structure, the system calculates the sum of numeric components of all rows in the internal table.

1. At operations
   * At First…. EndAt.
   * At Last …EndAt.
   * At New <fieldname>… EndAt .
   * At End OF <fieldname>….EndAt.
   * On Change Of <fieldname>…. End on. ( LHS rule does not apply rather if the field changes the statement gets executed.