A collection is an ordered group of elements having the same data type. Each element is identified by a unique subscript that represents its position in the collection.

PL/SQL provides three collection types:

* Index-by tables or Associative array
* Nested table
* Variable-size array or Varray

Oracle documentation provides the following characteristics for each type of collections:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Collection Type** | **Number of Elements** | **Subscript Type** | **Dense or Sparse** | **Where Created** | **Can Be Object Type Attribute** |
| Associative array (or index-by table) | Unbounded | String or integer | Either | Only in PL/SQL block | No |
| Nested table | Unbounded | Integer | Starts dense, can become sparse | Either in PL/SQL block or at schema level | Yes |
| Variable-size array (Varray) | Bounded | Integer | Always dense | Either in PL/SQL block or at schema level | Yes |

Index-By Table

An **index-by** table (also called an associative array) is a set of **key-value**pairs. Each key is unique and is used to locate the corresponding value. The key can be either an integer or a string.

An index-by table is created using the following syntax. Here, we are creating an index-by table named **table\_name** whose keys will be of *subscript\_type*and associated values will be of *element\_type*

TYPE type\_name IS TABLE OF element\_type [NOT NULL] INDEX BY subscript\_type;

table\_name type\_name;

Collection Methods

PL/SQL provides the built-in collection methods that make collections easier to use. The following table lists the methods and their purpose:

|  |  |
| --- | --- |
| **S.N.** | **Method Name & Purpose** |
| 1 | **EXISTS(n)** Returns TRUE if the nth element in a collection exists; otherwise returns FALSE. |
| 2 | **COUNT** Returns the number of elements that a collection currently contains. |
| 3 | **LIMIT** Checks the Maximum Size of a Collection. |
| 4 | **FIRST** Returns the first (smallest) index numbers in a collection that uses integer subscripts. |
| 5 | **LAST** Returns the last (largest) index numbers in a collection that uses integer subscripts. |
| 6 | **PRIOR(n)** Returns the index number that precedes index n in a collection. |
| 7 | **NEXT(n)** Returns the index number that succeeds index n. |
| 8 | **EXTEND** Appends one null element to a collection. |
| 9 | **EXTEND(n)** Appends n null elements to a collection. |
| 10 | **EXTEND(n,i)** Appends n copies of the ith element to a collection. |
| 11 | **TRIM** Removes one element from the end of a collection. |
| 12 | **TRIM(n)** Removes n elements from the end of a collection. |
| 13 | **DELETE** Removes all elements from a collection, setting COUNT to 0. |
| 14 | **DELETE(n)** Removes the nth element from an associative array with a numeric key or a nested table. If the associative array has a string key, the element corresponding to the key value is deleted. If n is null, DELETE(n) does nothing. |
| 15 | **DELETE(m,n)** Removes all elements in the range m..n from an associative array or nested table. If m is larger than n or if m or n is null, DELETE(m,n) does nothing. |

Collection Exceptions

The following table provides the collection exceptions and when they are raised:

|  |  |
| --- | --- |
| **Collection Exception** | **Raised in Situations** |
| COLLECTION\_IS\_NULL | You try to operate on an atomically null collection. |
| NO\_DATA\_FOUND | A subscript designates an element that was deleted, or a nonexistent element of an associative array. |
| SUBSCRIPT\_BEYOND\_COUNT | A subscript exceeds the number of elements in a collection. |
| SUBSCRIPT\_OUTSIDE\_LIMIT | A subscript is outside the allowed range. |
| VALUE\_ERROR | A subscript is null or not convertible to the key type. This exception might occur if the key is defined as a PLS\_INTEGER range, and the subscript is outside this range. |

Advantages

As for collections in PL/SQL (used as bulk processing buffers) - the SINGLE ONLY advantage is to reduce the number of context switches between the PL/SQL and SQL engines.  
  
Such a collection buffer can push or pull n rows between these two engines, which means less context switching ("row data pushing") needs to be performed between the two engines versus doing it 1 row at a time.