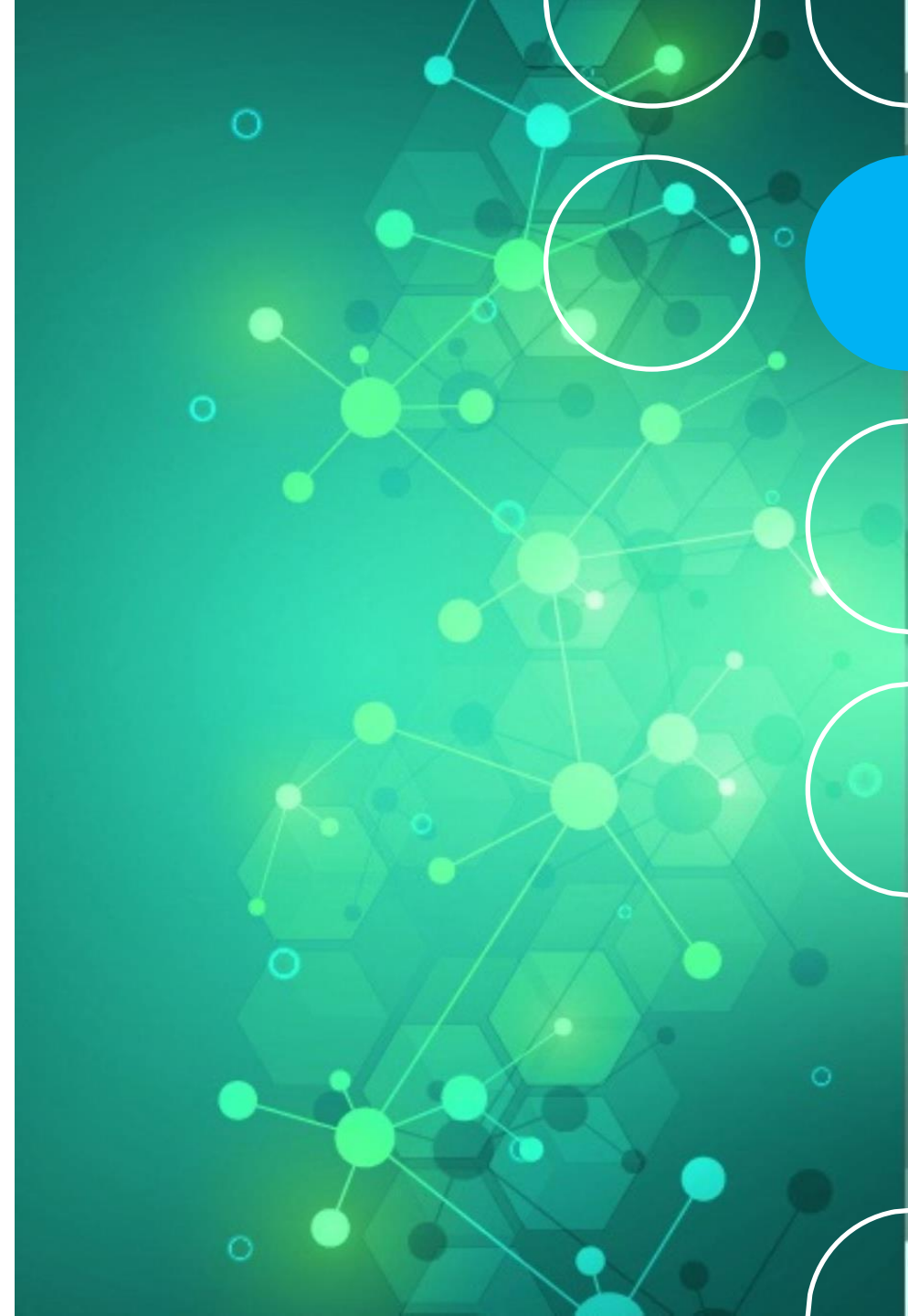


# ABAP

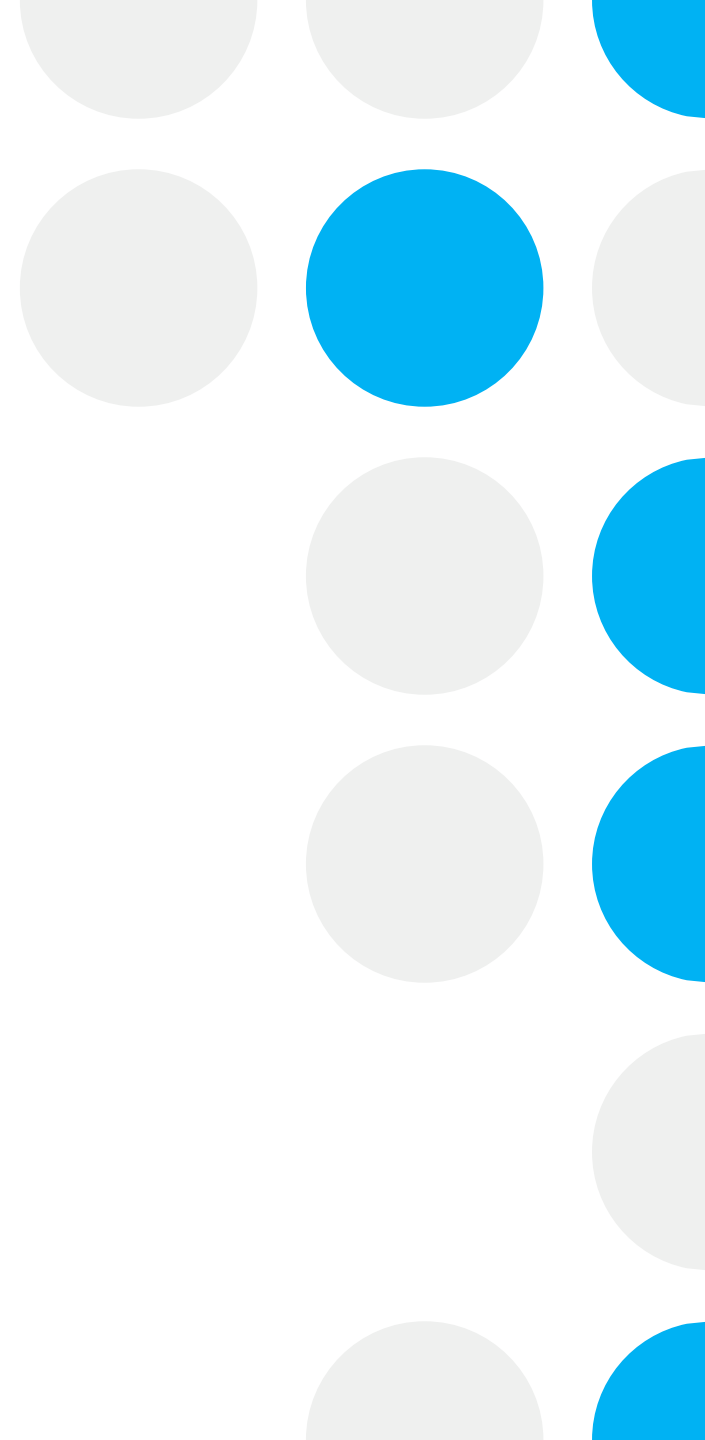
Interne Tabellen

---



# Einführung

- Es können **beliebig viele Zeilen** eines beliebigen Zeilentyps im Arbeitsspeicher abgelegt werden, deren Anzahl erst zur Programmlaufzeit festgelegt wird.
  - Zeilentypen können elementare Datentypen sein, Strukturen oder sehr komplexe dynamische Datenobjekte.
  - Auch Referenzvariablen können in internen Tabellen abgespeichert werden.
- 



# Definitionen

```
TYPES: BEGIN OF ts_schueler,  
        id          TYPE i,  
        name        TYPE c LENGTH 30,  
        vorname     TYPE c LENGTH 20,  
        klasse      TYPE c LENGTH 4,  
    END OF ts_schueler,  
    tt_schueler TYPE TABLE OF ts_schueler.
```

```
DATA: gs_schueler TYPE ts_schueler,  
      gt_schueler TYPE tt_schueler.
```

## Konventionen

*Erster Präfix:*

t für Type

*Zweiter Präfix:*

s für Struktur

t für Tabelle

# Tabelle füllen und ausgeben

```
gt_schueler = VALUE #(  
  ( id = 1  name      = 'Adam'      vorname = 'Tobias'  klasse = 'B12a')  
  ( id = 2  name      = 'Dorfmann'  vorname = 'Laura'   klasse = 'FS12')  
  ( id = 3  name      = 'Karl'      vorname = 'Max'     klasse = 'B12a')  
  ( id = 4  name      = 'Schuler'   vorname = 'Johannes' klasse = 'FS12')  
  ( id = 5  name      = 'Schuler'   vorname = 'Thomas'  klasse = 'B12a')  
  ( id = 6  name      = 'Zumm'     vorname = 'Laura'   klasse = 'FS12')  
)  
cl_demo_output=>display( gt_schueler ).
```

Klasse ***cl\_demo\_output*** kann für verschiedene Testausgaben verwendet werden  
Statische Methode ***display*** wird über => aufgerufen, Parameter kann interne Tabelle, Struktur oder einzelnes Feld sein

# Eine Zeile lesen

```
TRY.  
    gs_schueler = gt_schueler[ id = 2 ].  
    cl_demo_output=>display( gs_schueler ).  
    gs_schueler = gt_schueler[ id = 26 ].  
    cl_demo_output=>display( gs_schueler ).  
CATCH cx_root.  
    MESSAGE 'Nummer nicht vorhanden' TYPE 'S' DISPLAY LIKE 'E'.  
ENDTRY.
```

Lesen erfolgt über die Angabe des gewünschten Satzes in eckigen Klammern  
**TRY-CATCH** (mit Standardfehlerklasse *cx\_root*) ist erforderlich, da ein Programmabsturz erfolgt, wenn der angegebene Index nicht gefunden wird

# Mehrere Zeilen der Tabelle

```
DATA: gt_schueler_klasse TYPE tt_schueler.

LOOP AT gt_schueler INTO gs_schueler WHERE klasse = 'B12a'.
  APPEND gs_schueler TO gt_schueler_klasse.
ENDLOOP.

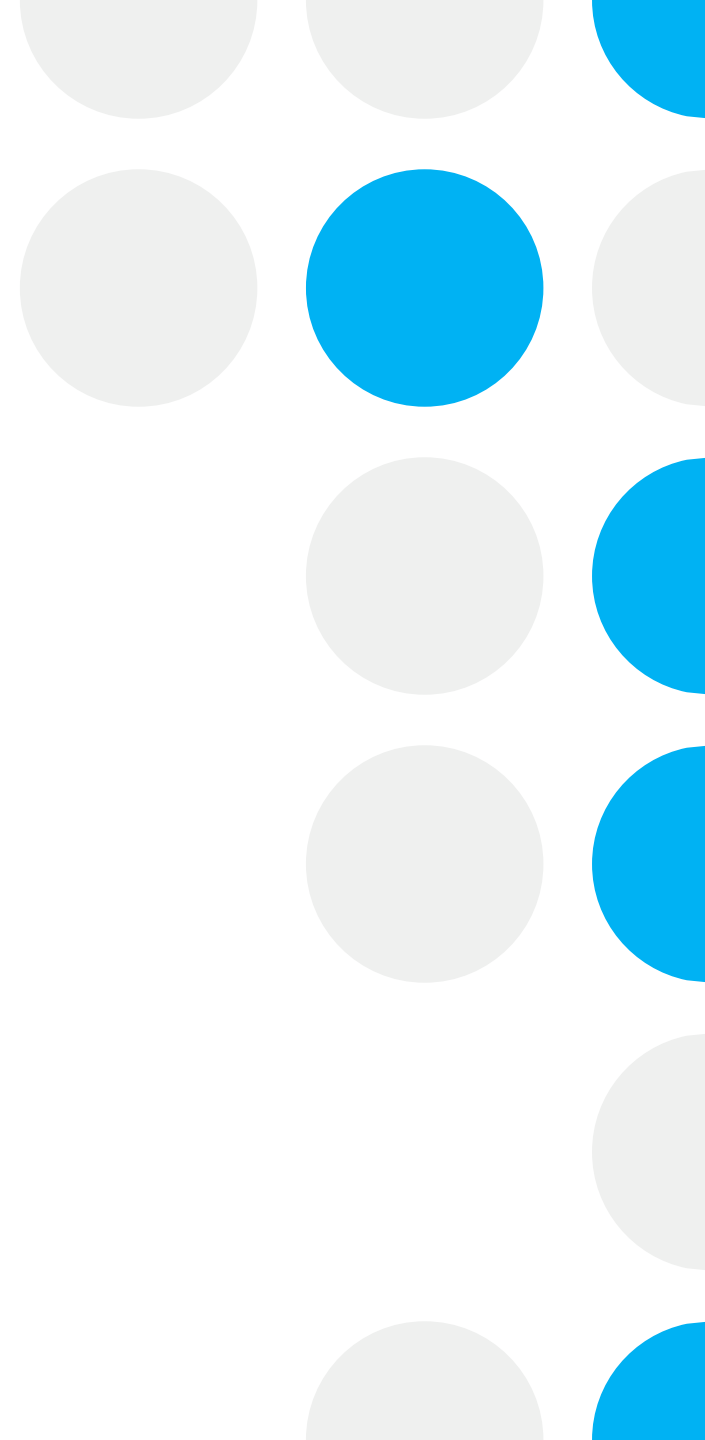
IF lines( gt_schueler_klasse ) > 0.
  cl_demo_output=>display( gt_schueler_klasse ).
ELSE.
  MESSAGE 'Klasse nicht vorhanden' TYPE 'S' DISPLAY LIKE 'E'.
ENDIF.
```

---

# Eine Zeile löschen

```
DELETE gt_schueler WHERE id = 1.  
cl_demo_output=>display( gt_schueler ).
```

---



# Neue Zeile anhängen

```
gs_schueler = VALUE ts_schueler(  
    id      = 7  
    name    = 'Chistof'  
    vorname = 'Emma'  
    klasse  = 'B12a'  
).  
APPEND gs_schueler TO gt_schueler.  
cl_demo_output=>display( gt_schueler ).
```

---



# Ändern einer Zeile

## Sortieren der Tabelle

```
gs_schueler = gt_schueler[ id = 3 ].  
gs_schueler-klasse = 'B11a'.
```

```
DATA(ln_id) = gs_schueler-id.  
MODIFY gt_schueler FROM gs_schueler INDEX ln_id.
```

```
SORT gt_schueler BY klasse name.  
cl_demo_output=>display( gt_schueler ).
```

---