SAP
table et structure interne

Zineb BOUGROUN

Plan

- Structure interne
- Table interne

Les structures

- **Qu'est-ce qu'une structure ?**

Une structure, c'est un ensemble de variables qui sont regroupées. Par exemple, imaginons que l'on veuille définir une personne. Chaque personne possède un nom, un prénom, et un age (pour faire simple). Alors au lieu de créer des variables nom, prénom, et âge, nous allons créer une structure personne, qui regroupera ces 3 variables.
Une structure, c'est donc une type complexe, c'est à dire qu'il ne stocke pas qu'une seule information, mais plusieurs.

Syntaxe :

Nous allons tout d'abord apprendre à créer un type structure. La syntaxe est :

```
TYPES begin of ty_personne,
        nom     TYPE c length 10,
        prenom  TYPE c length 10,
        age     TYPE i,
        end of ty_personne.
```

Nous venons de créer un nouveau type, une structure, qui possède 3 "champs" :

un champ **nom**, correspondant à une chaine de caractère de longueur 10

un champ **prenom**, correspondant à une chaine de caractère de longueur 10

un champ **age**, correspondant à un entier.

Les structures

```
TYPES begin of ty_personne,
                    nom      TYPE c length 10,
                    prenom  TYPE c length 10,
                    age      TYPE i,
                    end of ty_personne.
      DATA : v_personne1 TYPE ty_personne,
                v_personne2 TYPE ty_personne.

v_personne1-nom     = 'Grin'.
v_personne1-prenom = 'Alexandre'.
v_personne2-age      = 28.
v_personne2-nom     = 'Morisseau'.
v_personne2-prenom = 'Bertrand'.
v_personne2-age      = 30.
```

```
DATA wa_sflight TYPE sflight.

* en utilisant une table de dictionnaire


wa_sflight-carrid  =  'AA'.
wa_sflight-connid = '200'.
```

Définition des structures avec une référence de type du dictionnaire

- Vous pouvez définir des objets de données structurés (également appelés structures) dans ABAP. Ceci vous permet de combiner des variables correspondantes dans un seul objet. Les structures peuvent s'imbriquer. Ceci signifie que d'autres structures ou même des tables peuvent devenir des sous-objets de votre structure originale.

- Deux différents types de structure existent dans les programmes ABAP :
  - des structures définies par

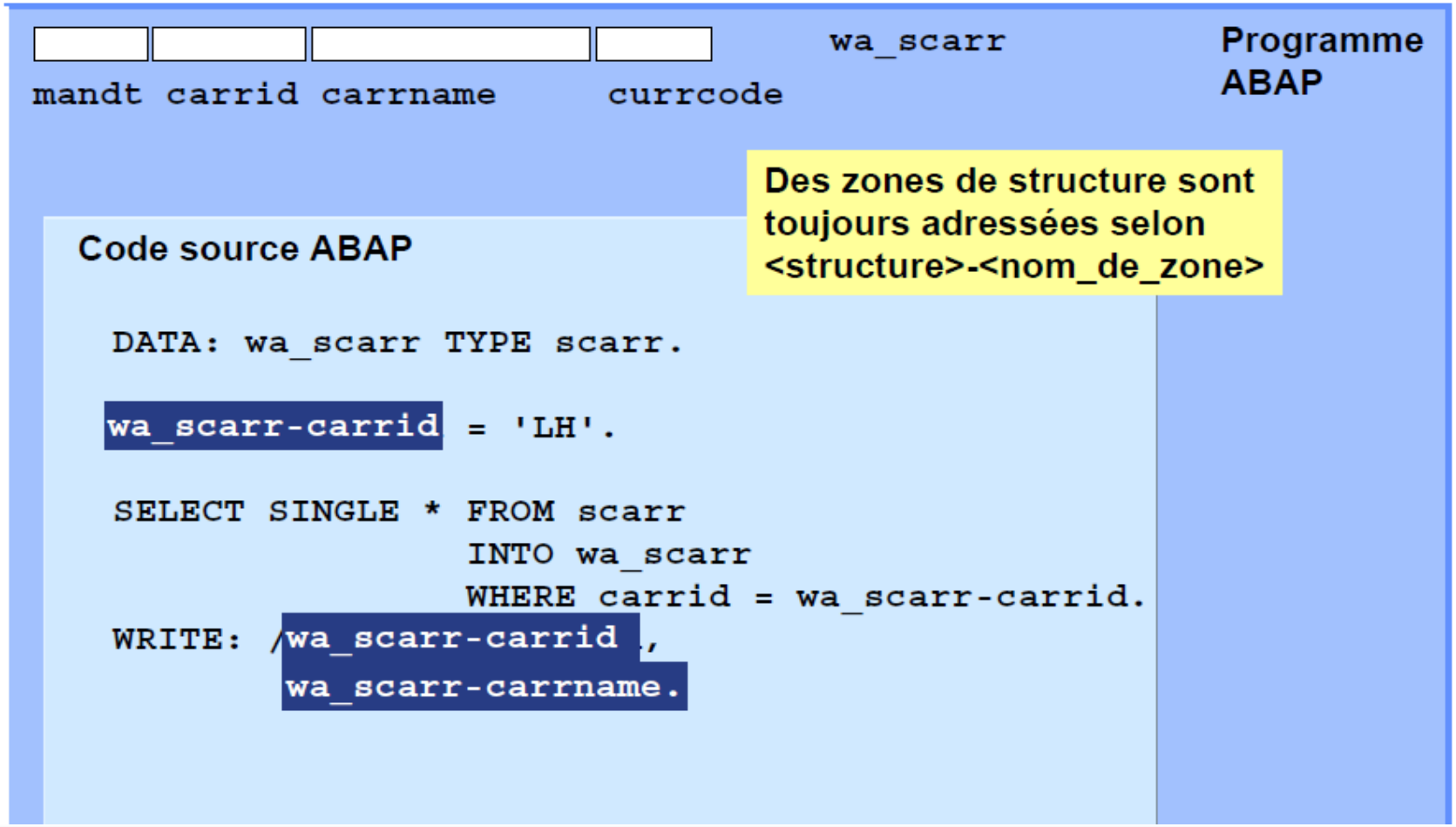
    DATA <name> TYPE <structure_type>.

    Ces types de structure servent de zones cibles pour les accès à la base de données ou pour les calculs exécutés localement dans le programme.
  - des structures définies par

    TABLES <ABAP-Dictionary-Structure>.

    Ces types de structure sont techniquement gérés dans leur propre domaine. Pour la version 4.0, les structures **TABLES doivent être utilisées uniquement comme des structures d'interface pour des** écrans.

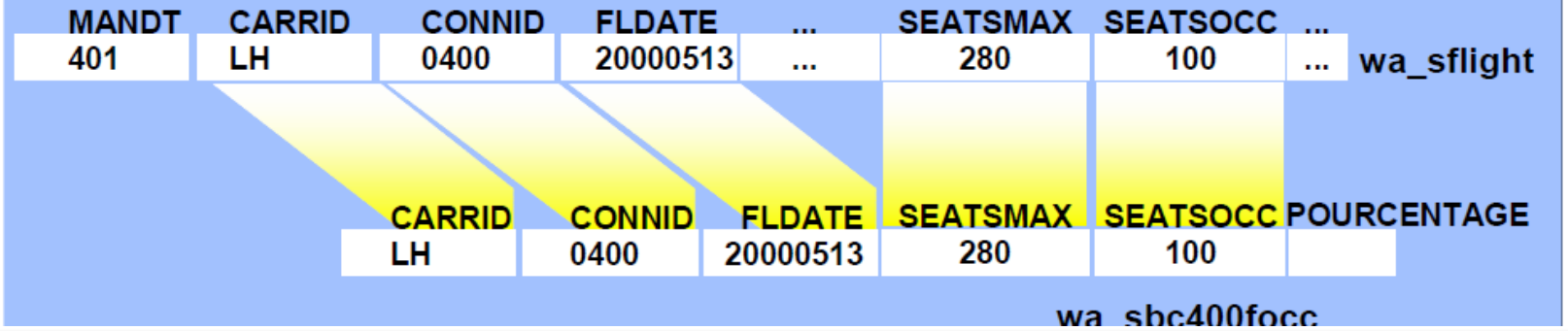
Adressage des zones d'une structure



Affectation de valeurs zone à zone

```
MOVE-CORRESPONDING <rec1> TO <rec2>.
```

```
DATA: wa_sflight TYPE sflight,
      wa_sbc400focc TYPE sbc400focc.
...
MOVE-CORRESPONDING wa_sflight TO wa_sbc400focc.
```



Les tables internes

Les tables internes sont l'équivalent en ABAP des tableaux (en C, C++), ou des Vecteurs (en Java).
Une table interne contient un nombre n d'enregistrements du même type, et la taille d'une table interne est dynamique (elle grossit en fonction du nombre d'enregistrements que l'on insère).

Syntaxe :

- Voici ci-dessous la déclaration d'une table interne contenant des enregistrements de type chaine de caractères de longueur 10.

- TYPES ty_prenom(10) TYPE c.

  DATA it_liste_prenom TYPE STANDARD TABLE OF ty_prenom.

Les tables internes

- **Une table de structures :**
- TYPES : begin of ty_personne,
  nom TYPE c length 10,
  prenom TYPE c length 10,
  age TYPE i,
  end of ty_personne.

- DATA it_personnes TYPE STANDARD TABLE OF ty_personne.
- Ici nous déclarons une table interne qui pourra contenir des enregistrement de type ty_personne, c'est à dire des structures.

  imaginons que nous voulons stocker des enregistrements de la table SFLIGHT, par exemple ceux de la compagnie American Airlines (CARRID = 'AA').
  Imaginons que nous voulions toutes les caractéristique d'un enregistrement, même si dans la pratique on ne prend que les champs qui nous intéressent.
  Au lieu de chercher tous les types des champs de la table SFLIGHT, nous pouvons simplement écrire :

- DATA : it_sflight TYPE STANDARD TABLE OF sflight.

Les tables internes

- **Faire référence à une donnée plutôt qu'à un type :**
- Dans la pratique, il est presque certain que nous aurons besoin de manipuler un enregistrement à un moment ou un autre, donc autant déclarer une variable de type structure ayant les caractéristique de SFLIGHT, comme cela :

Zone de texte

- DATA wa_sflight TYPE sflight.

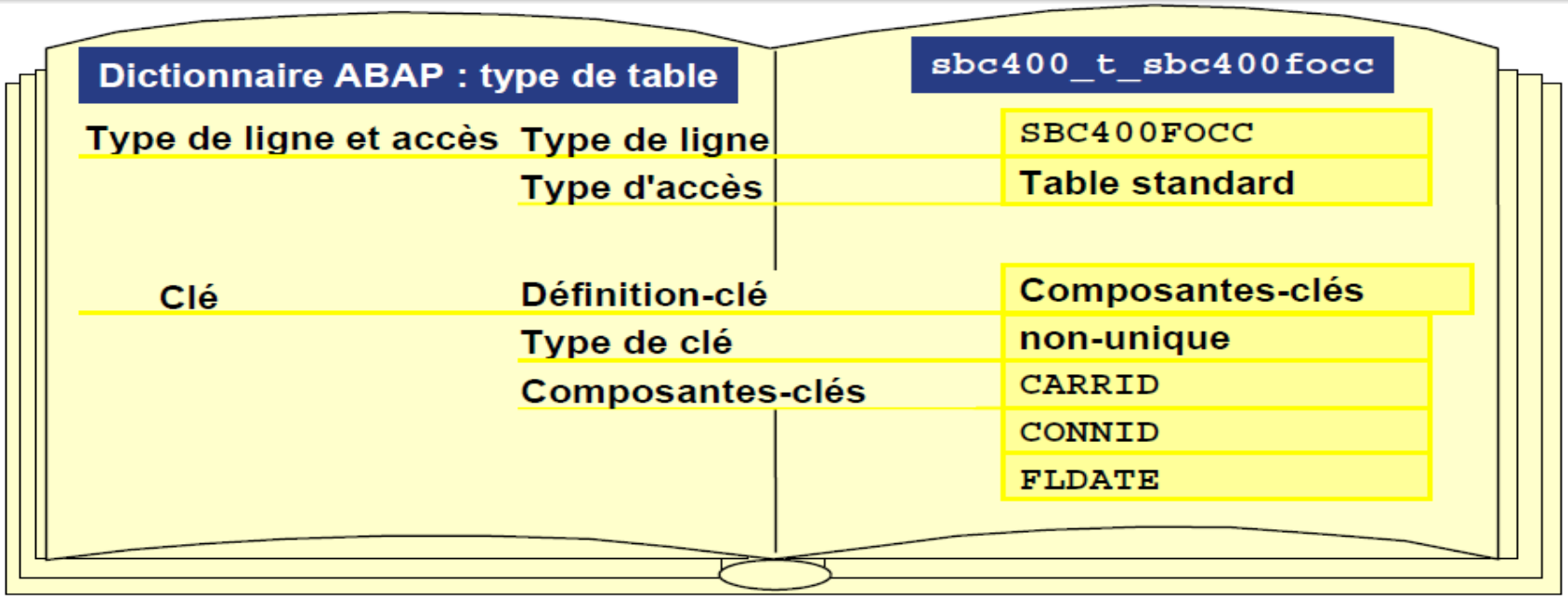
  DATA it_sflight TYPE STANDARD TABLE OF wa_sflight.

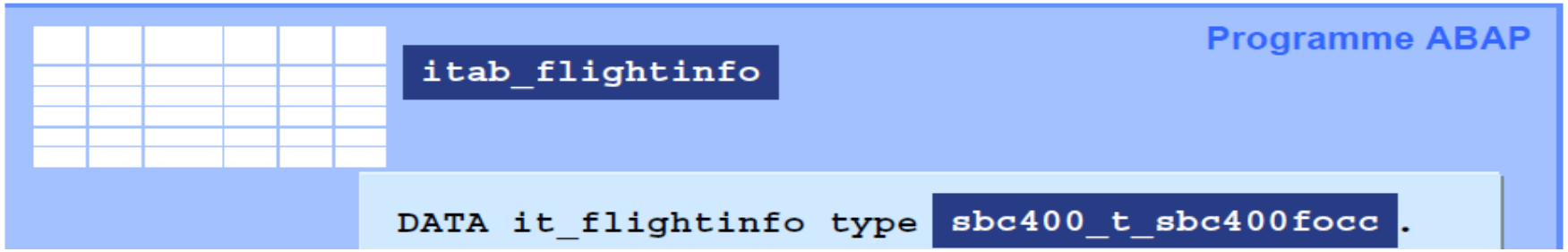
- Comme vous le voyez, il est tout à fait possible, au lieu de faire référence à un type, de faire référence à une donnée de ce type. J'essayerai de varier les déclarations afin que voyez différentes syntaxes au cours de ce tutorial

Type de table

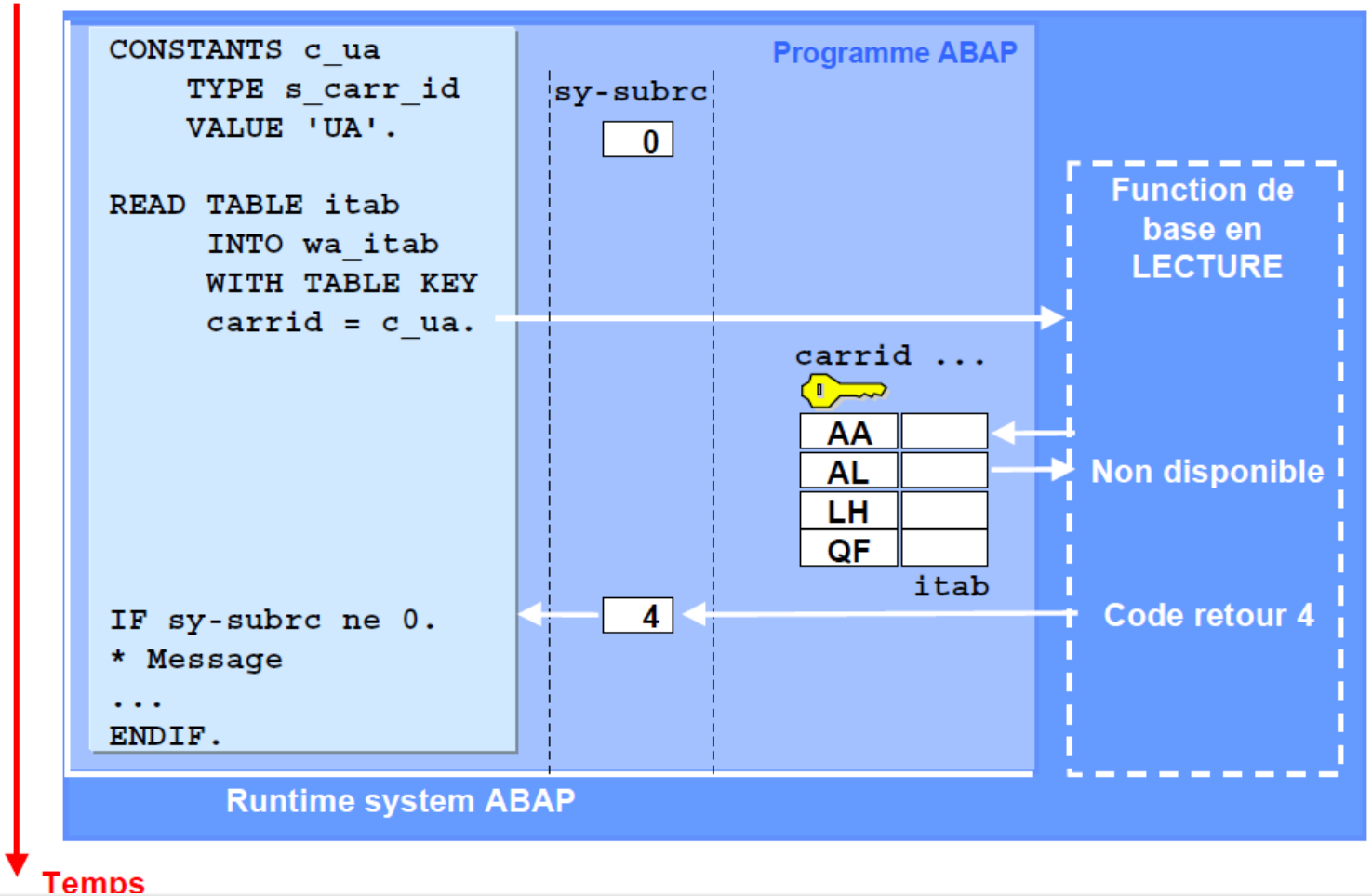
- Vous devez définir les informations suivantes afin de spécifier entièrement un type de table :
  - **Type de ligne : vous pouvez enregistrer les informations dans les colonnes requises, leurs noms et types, en** définissant un type de structure comme type de ligne.
  - **Clé : une clé correctement spécifiée doit définir quelles colonnes doivent être des colonnes-clé. Dans quel** ordre ? La clé doit-elle désigner un seul enregistrement de la table interne (clé unique) ? Les clés uniques ne peuvent pas être définies pour tous les types de table.
  - **Type de table : il existe trois types de table : les tables standard, les tables triées et les tables d'adresses** calculées.

Déclaration de tables internes en référence au Dictionnaire



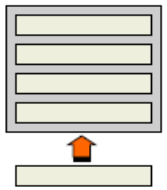


Codes retour d'instruction ABAP



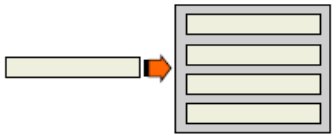
Traitement des enregistrements individuels

Ajouter



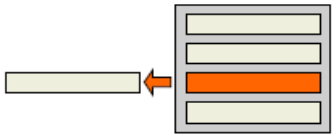
```
APPEND wa_itab to itab.
```

Insérer



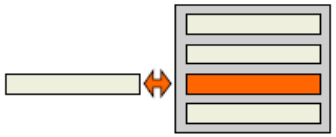
```
INSERT wa_itab INTO itab <condition>.
```

Lire



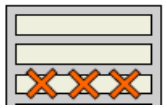
```
READ TABLE itab INTO wa_itab <condition>.
```

Modifier



```
MODIFY TABLE itab <condition>.
```

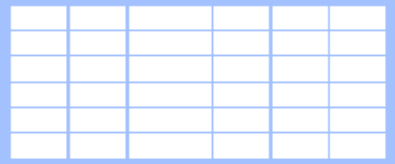
Supprimer



```
DELETE itab <condition>.
```

Exemple : remplir une table interne ligne par ligne

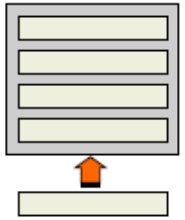
```
* Déclaration de table interne et d'espace de travail
DATA: it_flightinfo TYPE sbc400_t_sbc400focc.
DATA: wa_flightinfo TYPE sbc400focc.
```



it_flightinfo

wa_flightinfo

```
* Remplir une structure wa_flightinfo avec des valeurs
...
* Ajouter la structure wa_flightinfo dans une table
* interne
APPEND wa_flightinfo TO it_flightinfo.
```

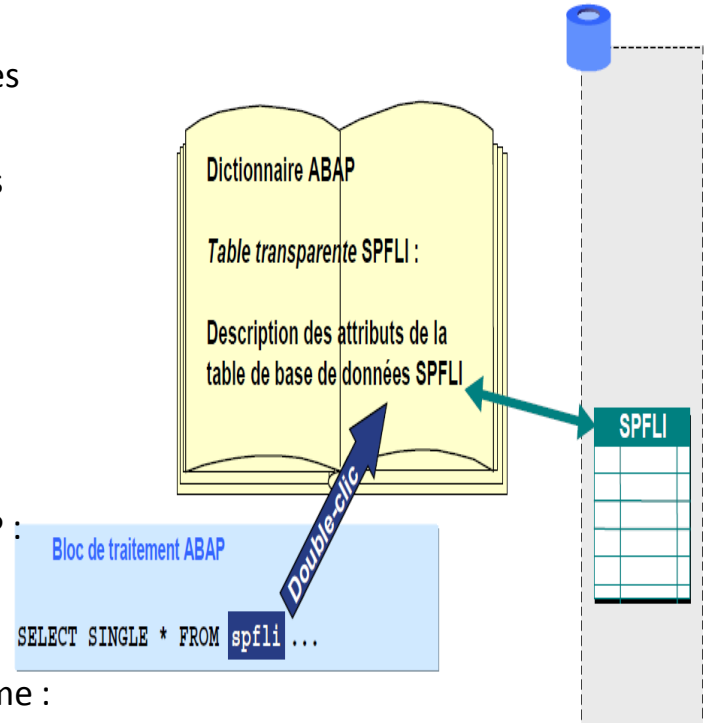


Traitement des enregistrements individuels

- Vous pouvez exécuter les opérations suivantes sur des enregistrements individuels de tables internes :
  - **APPEND ajoute le contenu d'une structure qui a le même type que la ligne d'une table interne.** Cette opération peut être utilisée uniquement avec des tables standard.
  - **INSERT insère le contenu d'une structure qui a le même type que la ligne de table interne. Pour** une table standard, cela provoque un APPEND ; pour une table triée, cela provoque une insertion à la bonne place ; pour une table d'adresses calculées, cela provoque une insertion suivant l'algorithme d'adresse calculée .
  - **READ copie le contenu d'une ligne de table interne vers une structure qui a le même type que** la ligne.
  - **MODIFY écrase une ligne de table interne avec le contenu d'une structure qui a le même type que** la ligne..
  - **DELETE supprime une ligne de table interne.**

Informations dans le Dictionnaire ABAP

- **Les tables de base de données sont administrées dans le Dictionnaire ABAP. Dans lequel vous** trouvez les informations actuelles sur les attributs techniques de la table de base de données.
  - **Table Transparente :** Existe avec la même structure dans le dictionnaire aussi bien que dans la base de données exactement avec les mêmes données et champs.
  - **Table pool (regroupée) :** Les tables pool sont des tables logiques qui doivent être affectées à un pool de tables lorsqu'elles sont définies. Fondamentalement utilisé pour stocker des données système et des données de personnalisation.
  - **Table de cluster:** les tables de cluster sont des tables logiques qui doivent être affectées à un cluster de table lorsqu'elles sont définies. Fondamentalement utilisé pour conserver les données d'application. Les tables de cluster peuvent être utilisées pour stocker les données de contrôle. Ils peuvent également être utilisés pour stocker des données temporaires ou des textes, tels que la documentation.
- Il existe différents moyens par lesquels vous pouvez naviguer vers les tables transparentes dans le Dictionnaire ABAP :
  - choisissez Outils->ABAP Workbench->Développement->Dictionnaire pour appeler le dictionnaire ABAP directement et insérer le nom de la table transparente dans la zone de saisie appropriée, ou
  - naviguez directement vers le Dictionnaire ABAP à partir de l'éditeur ABAP pendant le traitement du programme : vous pouvez double-cliquer sur le nom de la table transparente dans la clause **FROM de** l'instruction **SELECT.**



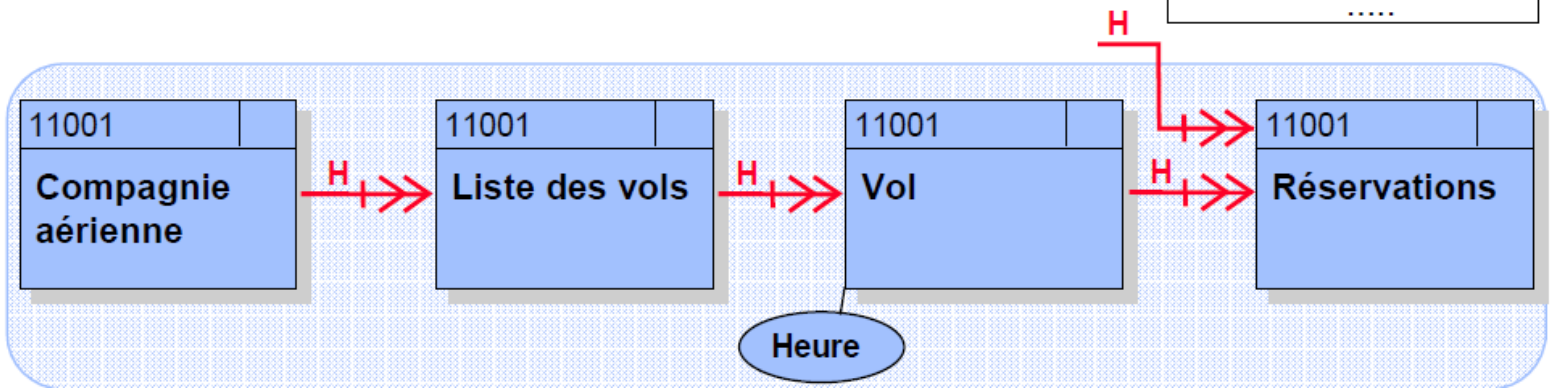
Exemple de tables

SCARR
Compagnie aérienne
MANDT
CARRID
CARRNAME
.....

SPFLI
Liste des vols
MANDT
CARRID
CONNID
AIRPFROM
AIRPTO
DEPTIME
.....

SFLIGHT
Vol
MANDT
CARRID
CONNID
FLDATE
SEATSMAX
SEATSOCC
.....

SBOOK
Réservations
MANDT
CARRID
CONNID
FLDATE
BOOKID
CUSTOMID
COUNTER
AGENCYNUM
.....

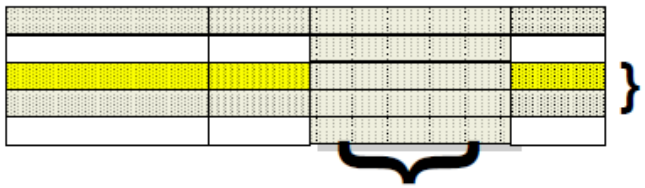


Lecture de tables de base de données

Quelles colonnes `SELECT <result>` `FROM <table>` Quelle(s) table(s) ?

`INTO <destination>` Vers quelles zones ?

`WHERE <condition>` Quelles lignes ?

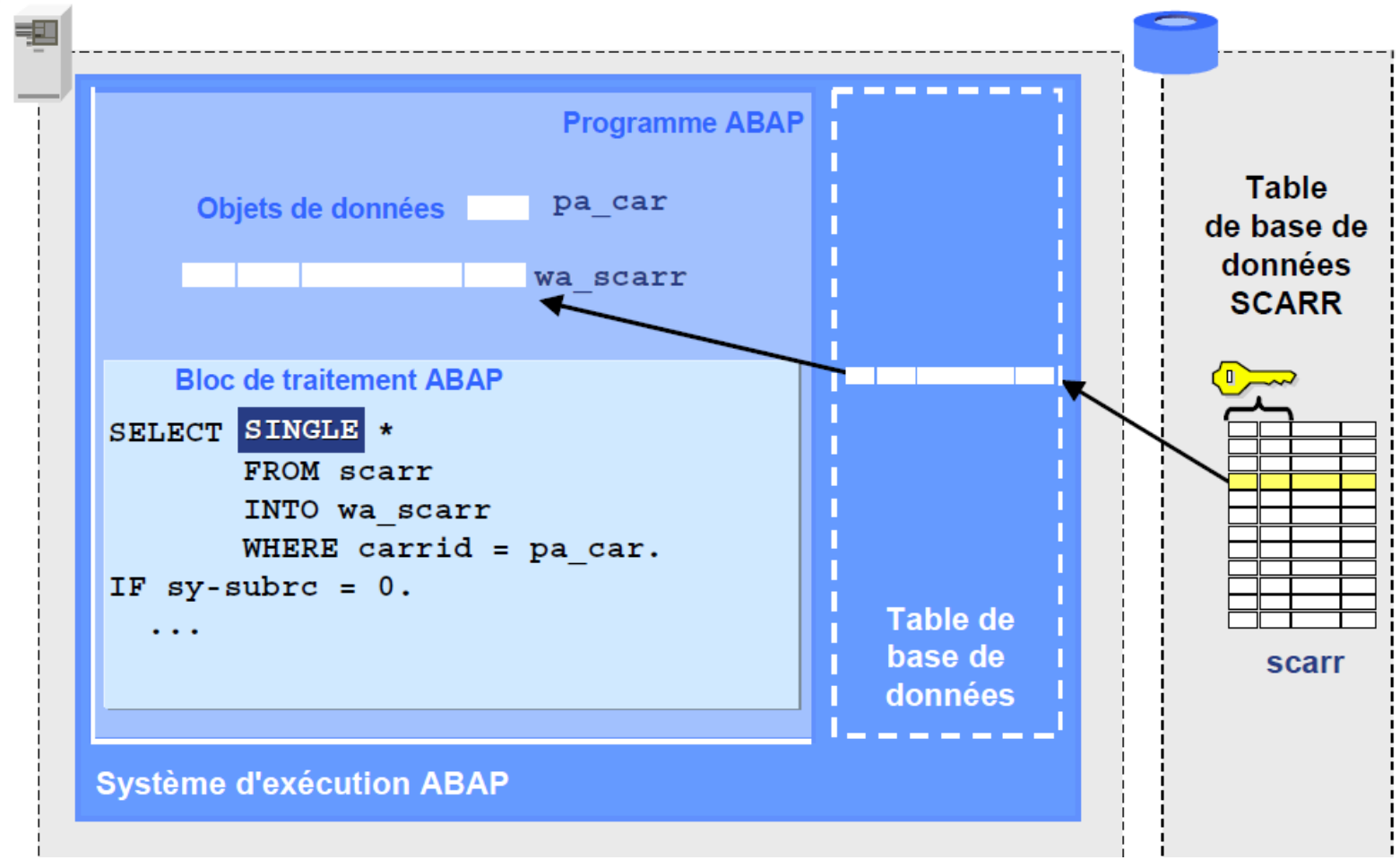


Ligne individuelle

Lignes multiples

Colonne spécifique

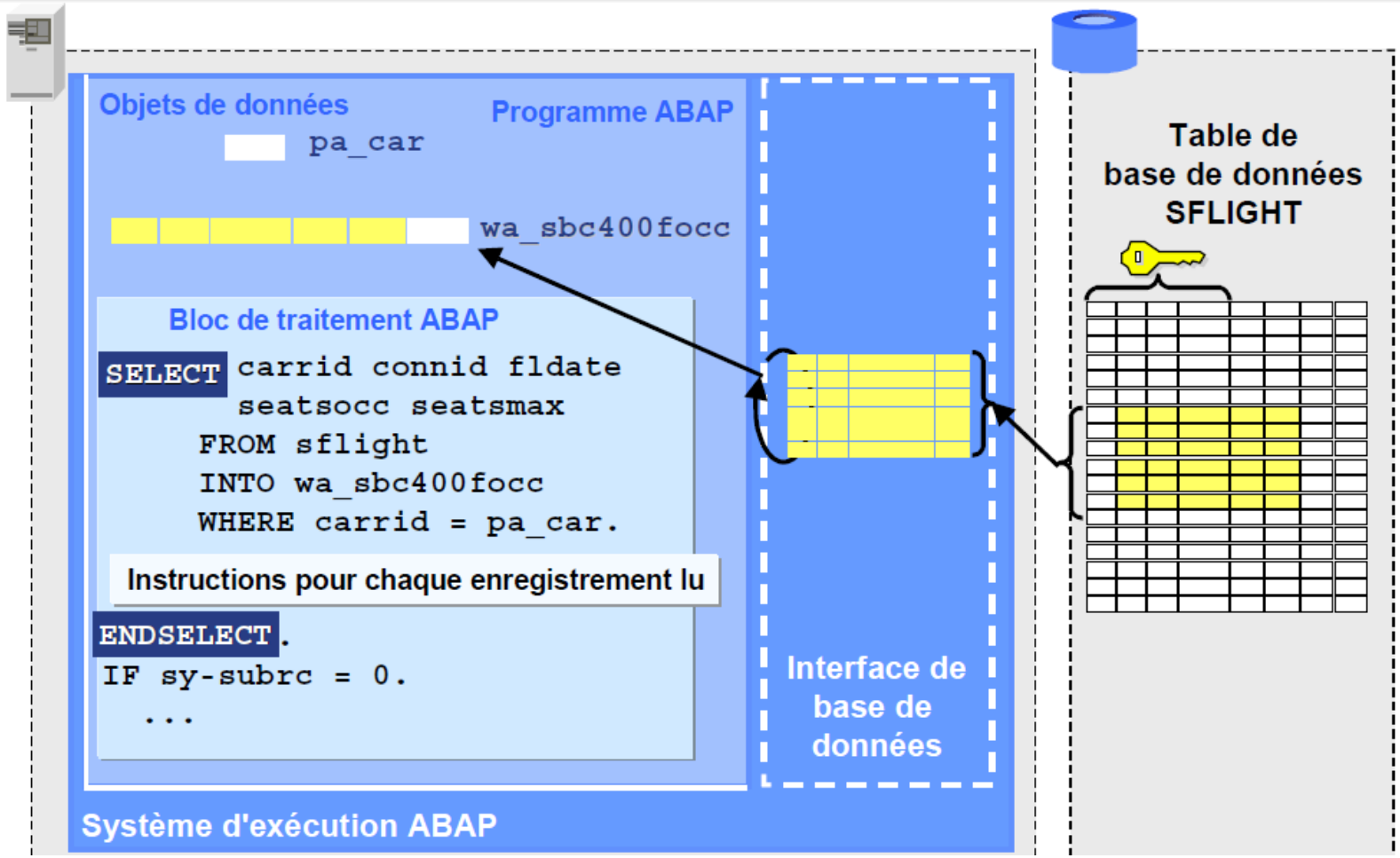
Traitement des enregistrements individuels



Traitement des enregistrements individuels

- L'instruction **SELECT SINGLE* vous permet de lire une ligne individuelle à partir d'une table de** base de données. Toutes les zones-clés doivent alors comporter la clause **WHERE. Elle informe** l'interface de base de données que toutes les colonnes appartenant à cette ligne doivent être lues. Si vous souhaitez uniquement une sélection spécifique de colonnes, vous pouvez y insérer une structure.
- Vous devez introduire le nom de la structure dans laquelle vous désirez que l'interface de base de données copie un enregistrement de données après la clause **INTO. Cette structure doit être** identique à celle des colonnes de la table, qui sont lues et alignées à gauche.
- Si vous utilisez l'option **CORRESPONDING FIELDS OF dans la clause INTO, vous pouvez** remplir l'espace de travail cible composante par composante. Le système ne retient que celles qui portent le même nom que les colonnes de la table. Si vous n'indiquez pas l'option, il complète l'espace de travail à partir de la gauche, indépendamment de sa structure.
- Si le système trouve l'entrée de table répondant aux conditions spécifiées, **SY-SUBRC adopte la** valeur 0.
- L'option **SINGLE informe la base de données qu'une ligne individuelle doit être lue. La base de** données termine la recherche une fois cette ligne trouvée. **SELECT SINGLE apporte donc une** meilleure performance d'accès à un enregistrement individuel qu'une boucle **SELECT, si vous** fournissez les valeurs de toutes les zones-clés.

Boucles SELECT

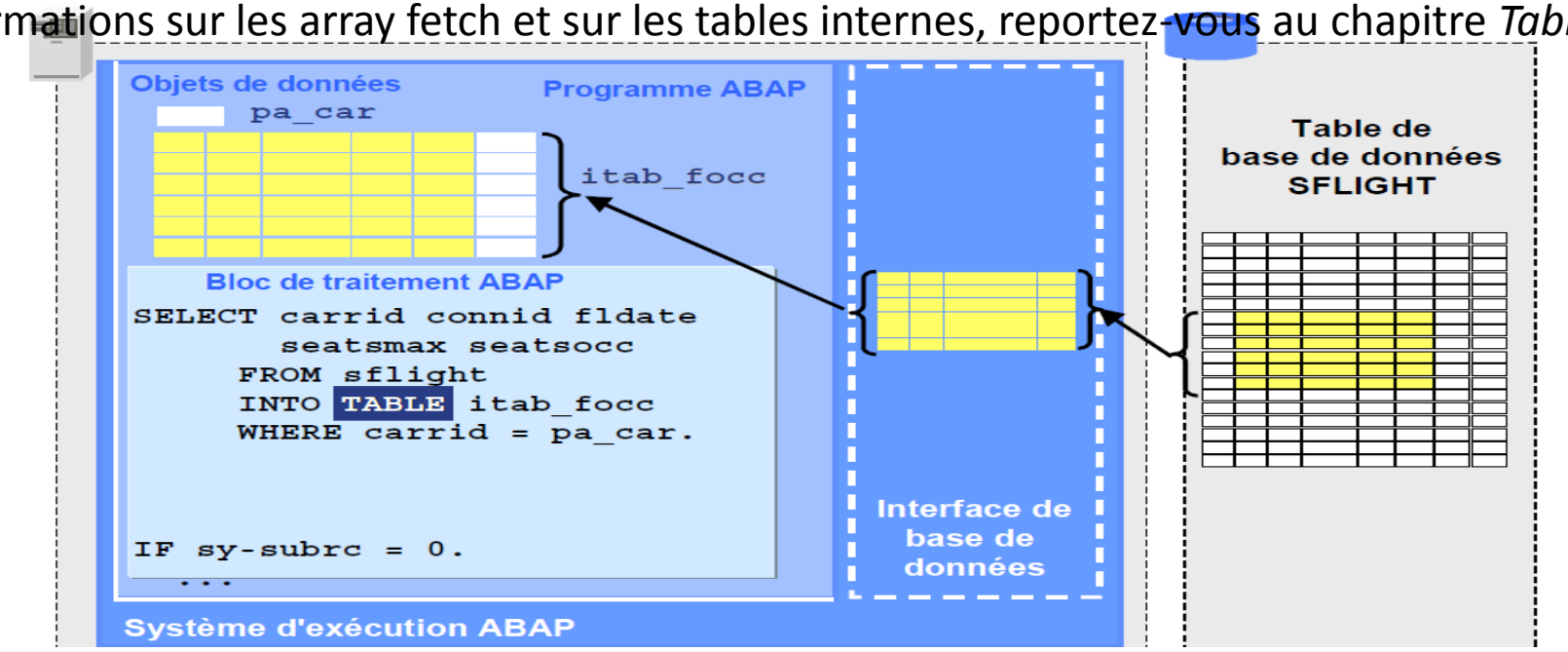


Boucles SELECT

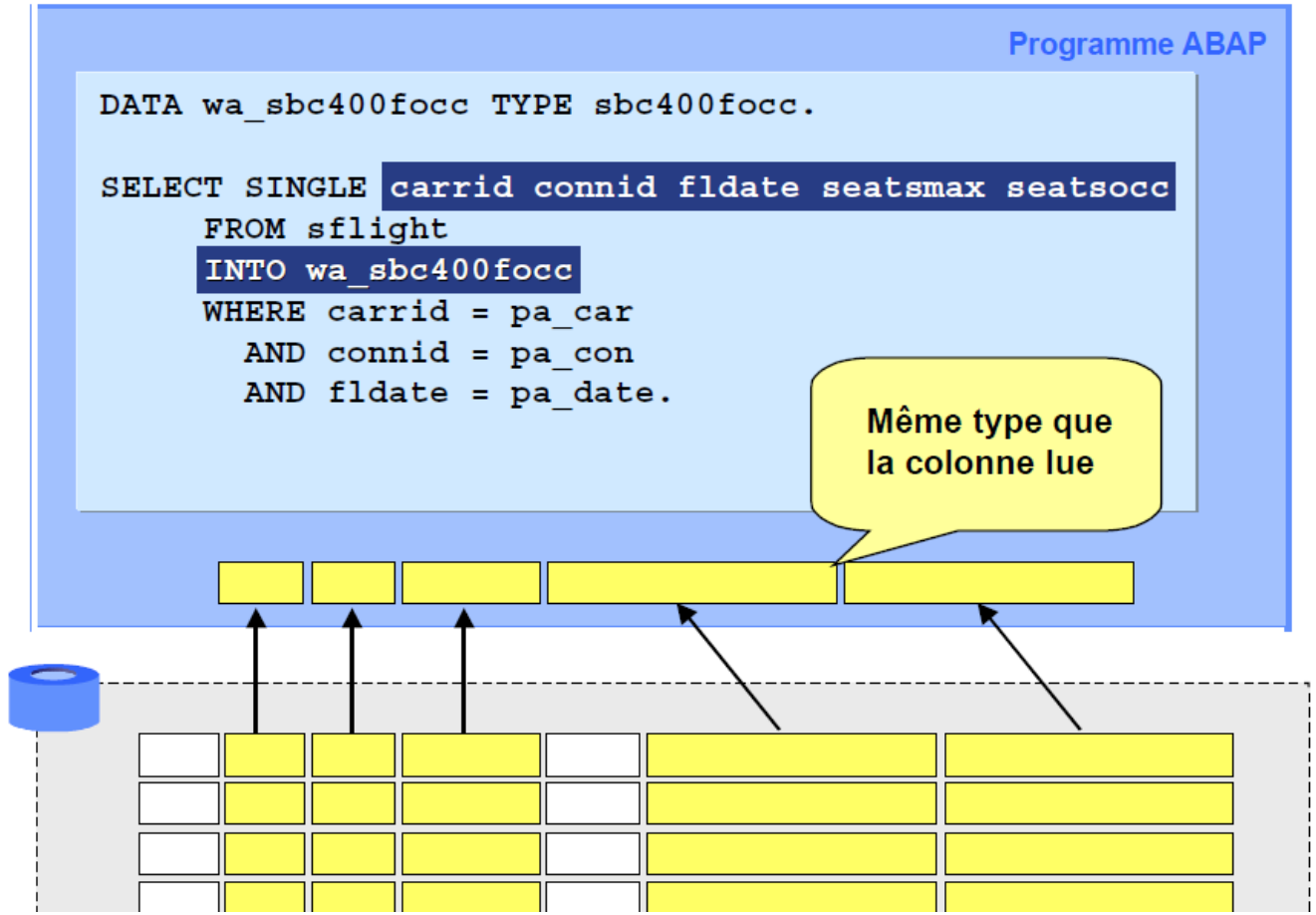
- Si vous n'utilisez pas l'option **SINGLE dans l'instruction SELECT, le système lit plusieurs** enregistrements multiples de la base de données. La liste des zones détermine les **colonnes dont les** données doivent être lues.
- Le nombre de **lignes à lire peut être limité par la clause WHERE. Elle effectue des restrictions soit** selon les zones-clés de la table de base de données, soit selon un index secondaire.
- De multiples conditions logiques peuvent être ajoutées dans la clause **WHERE en utilisant AND ou OR.**
- La base de données envoie les données à l'interface sous forme de piles. Le système d'exécution ABAP copie ligne par ligne les enregistrements de données dans la zone cible en utilisant une boucle. Il effectue également le traitement séquentiel de toutes les instructions situées entre **SELECT** et **ENDSELECT.**
- **SY-SUBRC = 0, lorsque le système a pu sélectionner au moins une entrée. Une fois l'instruction SELECT exécutée dans chaque passage de boucle, la zone de système SY-DBCNT contient le** nombre de lignes lues. Après l'instruction **ENDSELECT, il contient le nombre total de lignes lues.**

Accès à des tableaux

- L'option **INTO TABLE <itab> copie le contenu de l'interface de base de données dans la table** interne **itab. Cette opération s'appelle *ARRAY FETCH.***
- Comme les **ARRAY FETCH ne constituent pas une boucle, aucune instruction ENDSELECT n'est** utilisée.
- **SY-SUBRC = 0, lorsque le système a lu au moins une entrée de table.**
- Pour plus d'informations sur les array fetch et sur les tables internes, reportez-vous au chapitre *Tables internes.*



Clauses INTO



Clauses INTO

- La clause **INTO spécifie les objets de données dans lesquels vous souhaitez placer les données de la** table. Vous pouvez :
  - définir dans votre programme une **structure plate qui comporte les zones dans le même ordre que** la liste des zones de la clause **SELECT. Ensuite, indiquez le nom de la structure dans la clause INTO. Le contenu est copié par position. Les noms de zones de structure ne sont pas pris en** compte.
  - indiquer un ensemble d'**objets de données individuels dans la clause INTO. Par exemple :**
    - **DATA:**

      gd_carrid TYPE sflight-carrid,

      gd_connid TYPE sflight-connid,

      gd_fldate TYPE sflight-fldate,

      gd_seatsmax TYPE sflight-seatsmax,

      gd_seatsocc TYPE sflight-seatsocc.

      START-OF-SELECTION.

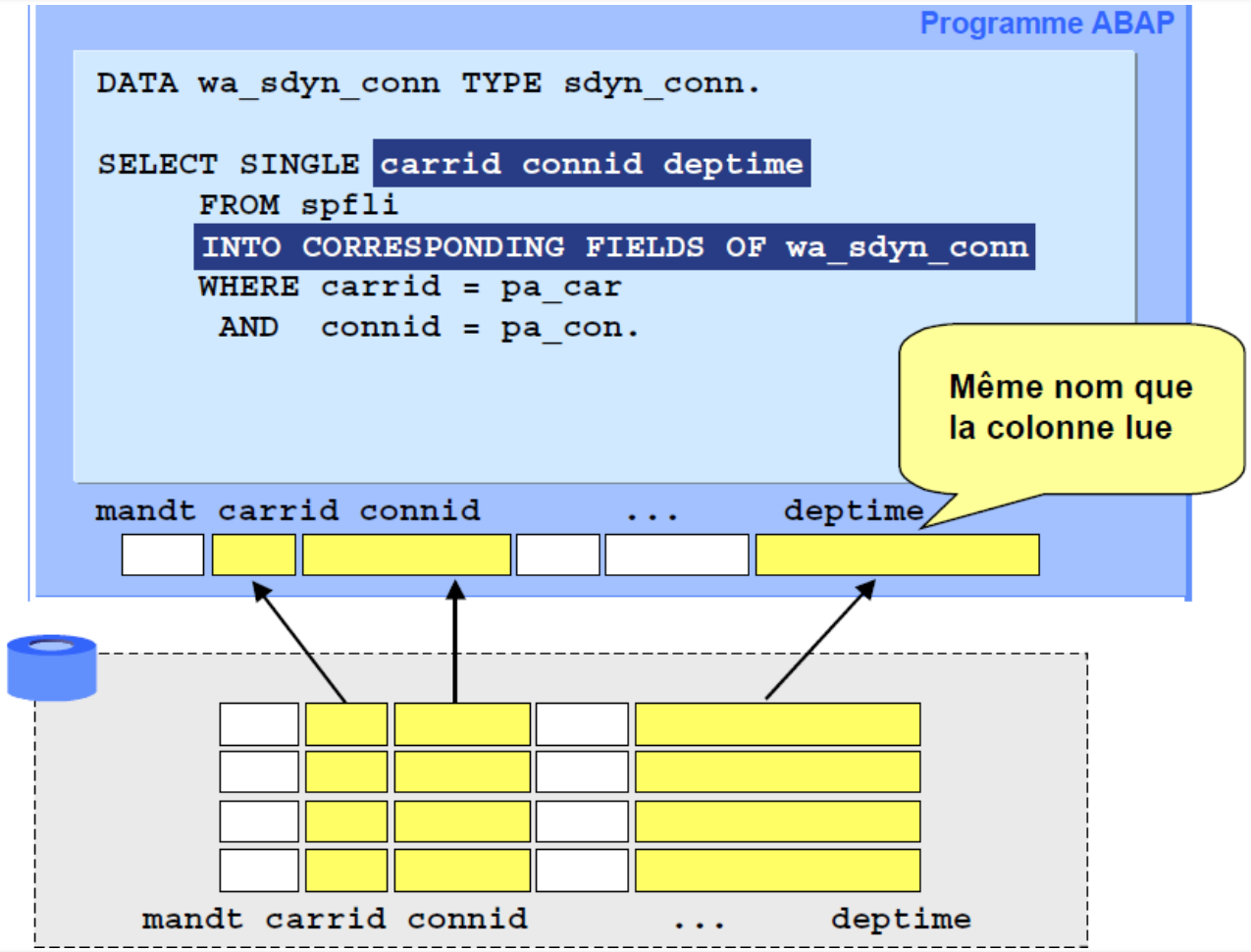
      SELECT carrid connid fldate seatsmax seatsocc

      FROM sflight

      INTO (gd_carrid, gd_connid, gd_fldate, gd_seatsmax, gd_seatsocc)

      WHERE ...

INTO CORRESPONDING FIELDS



INTO CORRESPONDING FIELDS

- Si vous utilisez la clause **INTO CORRESPONDING FIELDS, les données sont placées dans les** zones de structure du même nom.
- Avantages de cette construction :
  - la structure ne doit pas être structurée comme la liste des zones et ne doit pas être alignée à gauche;
  - elle est facile à gérer, car l'extension de la liste des zones n'exige pas d'autres modifications du programme, pour autant qu'une zone de la structure a le même nom et type.
- Désavantages de cette construction :
  - **INTO CORRESPONDING FIELDS est plus intensif au moment de l'exécution que INTO. La** durée d'exécution peut par conséquent se prolonger.
- Si vous souhaitez placer des données dans des colonnes de table interne de même nom en utilisant une création de tableaux, utilisez **INTO CORRESPONDING FIELDS OF TABLE <itab>.**

Exercices

- Réaliser un programme qui affiche la compagnie aérienne « American Airlines » sachant que la table des compagnies aérienne est « SCARR »
- Utiliser les champs CARRID CARRNAME CURRCODE et URL