

FICHE COBOL

ENTÊTE D'UN PROGRAMME COBOL :

IDENTIFICATION DIVISION.

PROGRAM-ID. [Nom du programme].

ENVIRONMENT DIVISION.

INPUT-OUTPUT SECTION.

FILE-CONTROL.

SELECT F ASSIGN TO DATABASE [Nom du fichier dans le système]

[ORGANIZATION type-organisation] ⇒ Choix du type du fichier (RELATIVE / INDEXED)

[ACCESS MODE type-d'accès] ⇒ Type d'accès direct (si fichier relative ou indexed)

[définition de clé] ⇒ **Donner le champ du fichier faisant office de clé.**

Si le fichier est séquentiel, on ne met que le SELECT

DATA DIVISION.

FILE SECTION.

FD F.

01 ENR.

COPY DDS-ENRCLI OF [Nom du fichier dans le système]. ⇒ **Crée une structure implémentée par le fichier**

WORKING-STORAGE SECTION.

77 [nom de variable] PIC 9(5). ⇒ **Déclaration d'une variable type numérique**

77 [nom de variable] PIC X(5). ⇒ **Déclaration d'une variable type texte**

77 [nom de variable] PIC X(6) VALUE "DUPOND". ⇒ **Seulement pour les constantes**

01 [Nom de la structure].

02 [Nom d'une variable] PIC [Type].

02 FILLER PIC X VALUE "-". ⇒ **Variable sans nom purement syntaxique**

...

02 [Nom de sous structure].

03 [Nom d'une variable] PIC [Type].

...

01 TABLEAU

05 CASE PIC 99 OCCURS 20. ⇒ **Création d'un tableau de 20 cases de type numérique à 2 valeurs.**

10 COLUMN PIC 99 OCCURS 20. ⇒ **Création de 20 colonnes dans le tableau.**

PROCEDURE DIVISION.

PRINCIPAL SECTION.

DEBUT.

[Corps de la section]

PERFORM TRAVAIL ⇒ **Appelle le sous programme travail**

FIN.

STOP RUN.

TRAVAIL SECTION. ⇒ **Sous programme appelé « travail ».**

DEBUT.

[Corps de la section]

FIN.

EXIT

INSTRUCTIONS :

- MOVE [Valeur] TO [Nom de variable] ⇒ **Affectation**
- IF [Condition]
 [Instructions]
ELSE
 [Instructions]
END-IF.
- PERFORM UNTIL [condition]
 [instructions]
END-PERFORM.

- TQ. \Rightarrow **Zone A**
 IF [condition] GO TO FTQ END-IF. \Rightarrow Condition d'arrêt
 [instructions]
 GO TO TQ.
 FTQ. \Rightarrow **Zone A**

LES CONDITIONS :

NOT \Rightarrow [!=] / AND / OR / NOT
 Exemple : IF A (< (B AND C)) OR = D

ARITHMETIQUE :

ADD I J GIVING K. \Rightarrow [K = I+J]
 ADD I TO K. \Rightarrow [K = K+I]
 SUBTRACT I FROM J GIVING K. \Rightarrow [K = J-I]
 SUBTRACT I FROM J. \Rightarrow [J = J-I]
 MULTIPLY I BY J GIVING K. \Rightarrow [K = I*J]
 MULTIPLY 2 BY J. \Rightarrow [J = J*2]
 DIVIDE 2 INTO J. \Rightarrow [J = J/2]
 DIVIDE I INTO J GIVING K \Rightarrow [K = J/I] REMAINDER L. \Rightarrow [L = J%I]

REDEFINES

01 DRAPEAU.
 05 TOUT PIC X(15) VALUE "BLEU BLANCROUGE".
 01 COULEURS REDEFINES DRAPEAU.
 05 GAUCHE PIC X(5).
 05 CENTRE PIC X(5)
 05 DROITE PIC X(5).

REDEFINES permet de séparer un seul champ en plusieurs

FICHIERS

INPUT (lecture) \Rightarrow READ \Rightarrow **Lit la ligne suivante dans un fichier**
 OUTPUT (création) \Rightarrow WRITE
 EXTEND (ajout séquentiel) \Rightarrow WRITE
 I-O (mise à jour) \Rightarrow READ puis REWRITE ou DELETE
 START F KEY[= ; > ; <...] [Variable clé] OF [Nom du fichier] \Rightarrow **Permet d'accéder, selon le signe arithmétique, au fichier a partir de l'endroit précisé**

[READ / WRITE / DELETE/ REWRITE/ START] INVALID KEY \Rightarrow **entrée dans INVALID KEY si la clé fichier est inexistante.**

EXEMPLE DE LECTURE D'UN FICHIER SEQUENTIEL

DEBUT.
 OPEN INPUT F. \Rightarrow **Ouverture du fichier en lecture.**
 MOVE "FAUX" TO EOF.
 READ F AT END MOVE "VRAI" TO EOF END-READ. \Rightarrow **Vérifie si le fichier est vide**
 PERFORM UNTIL EOF = "VRAI"
 [Traitement de l'enregistrement]
 READ F AT END MOVE "VRAI" TO EOF END-READ \Rightarrow **Vérifie quand le fichier se termine**
 END-PERFORM.
 FIN.
 CLOSE F.
 STOP RUN.

EXEMPLE DE CREATION D'UN FICHIER SEQUENTIEL

```
DEBUT.  
  OPEN OUTPUT F. ⇒ Ouverture d'un fichier en création. (Écrasement du fichier si existant)  
  ACCEPT ENR.  
  PERFORM UNTIL ENR = SPACES  
    WRITE ENR END-WRITE ⇒ Écrit le contenu de la variable ENR dans le fichier F  
    ACCEPT ENR  
  END-PERFORM.  
FIN.  
CLOSE F.  
STOP RUN.
```

COMPOSITION D'UN FICHIER SUR AS400

```
R ENRSTOCK  
  CODEART      15  
  LIBELLE      50  
  QTE          5      0  
  PU           5      2  
K CODEART
```

UNIQUE ← Signifie que la clé fichier est unique

Premier chiffre : Nombre de caractères pour la variable
Deuxième chiffre : Nombre de chiffre après la virgule

Indique le champ clé du fichier

SQL EN COBOL

```
EXEC SQL  
  INCLUDE SQLCA ⇒ Obligatoire à l'utilisation d'SQL en COBOL  
END-EXEC.
```

```
EXEC SQL  
  [une seule commande ou instruction]  
SQL END-EXEC.
```

Variables COBOL dans les instructions SQL élémentaire :

77 X PIC X(50). ---> :X

Champ de structure :

```
01 STRU.  
  05 A PIC X(50). ---> :STRU.A  
  05 B.  
    10 C PIC X(50). ---> :B.C !!!!! ⇒ :STRU.C invalide & :STRU.B.C invalide
```

```
EXEC SQL  
  SELECT NOM, PRENOM INTO :NOM.:PRENOM FROM ETUDIANT WHERE CODETU = 12345  
END-EXEC  
⇒ Copie des champs Nom Prenom dans les variables NOM PRENOM du programme COBOL
```

:[Nom d'une variable] ⇒ Peut être insérer dans une requête SQL

```
IF SQLCODE < 0  
  PERFORM ERREUR  
END-IF.
```

Si SQL CODE est à 100, il n'y a pas eu de résultats a la requête

SQL EN COBOL : LES CURSEURS

Traitement d'un curseur cobol :

```
EXEC SQL
    DECLARE [Nom du curseur] CURSOR FOR [Requête SQL]
END-EXEC
EXEC SQL
    OPEN [Nom du curseur]
END-EXEC.

EXEC SQL
    FETCH C2 INTO :CLIENT ⇒ Engage le curseur dans la structure CLIENT
END-EXEC.

PERFORM UNTIL SQLCODE = 100 ⇒ Verifie si il reste des valeurs dans CLIENT
    [Traiter les informations]
    EXEC SQL
        FETCH C2 INTO :CLIENT ⇒ Passe à la donnée suivante de CLIENT.
    END-EXEC
END-PERFORM.

EXEC SQL
    CLOSE [Nom du curseur]
END-EXEC.
```

FONCTION UTILISABLES AVEC SQL :

Chaînes :

ASCII(S) : code ASCII du premier caractère de s
CHAR(I) : conversion d'un nombre en chaîne
CONCAT(S1, S2) : concaténation des chaînes S1 et S2
DECIMAL(S) : conversion d'une chaîne en nombre
LENGTH(S) : donne la longueur d'un champ
LEFT(S, L) : les L premiers caractères de s
LOWER(S) (ou LCASE(S)) : conversion en minuscules
POSSTR(S1, S2) : retourne la position de chaîne2 dans chaîne1
REPLACE(S, S1, S2) : remplace S1 par S2 dans S
STRIP(S) ou TRIM : Supprime les espaces à gauche et à droite
STRIP(S, TRAILING) : à droite, LEADING à gauche
SUBSTR(S, deb, longueur) ou SUBSTRING: sous-chaîne.

Numériques :

+ - * /
ABS(I) : valeur absolue
RAND() : nombre aléatoire < 1
INT(I) : partie entière
ROUND(I, J) : arrondi de I avec J décimales

Date et Heure :

NOW() : retourne la date et l'heure courante
CURRENT DATE : date du jour
DATE(d), YEAR(d), MONTH(d), DAY(d), HOUR(d), MINUTE(d), SECOND(d), MONTHNAME(d), DAYNAME(d)
d + n YEAR(S) + n MONTH(s) + n DAY(s)
days(d1) – days(d2) : nombre de jours entre deux dates