Introduction :

Les stages d'entreprise, pions essentiels dans toutes formations professionnelles, sont très valorisés à la Faculté des Sciences et Techniques **(FST)** qui les considère comme des compléments indispensables à un enseignement théorique de haut niveau.

En ce qui me concerne, attirés par le fonctionnement des grandes entreprises j'ai opté pour MARSA MAROC vu qu'elle représente un espace propice pour l'épanouissement de La personnalité et à l'élargissement des connaissances techniques d'un étudiant.

Ceci est dû à plusieurs raisons telles que l'importance de MARSA MAROC et son positionnement à l'échelle nationale et internationale, ses équipements, et la qualité de ses services et de ses employés.

J'essayerai d'abord de montrer l'entreprise sous ses différentes facettes c'est-à-dire ses origines, les différents activités et services qu'elle offre mais surtout sa hiérarchisation et son fonctionnement avant de voir par la suite comment s'est déroulé mon stage en montrant les composantes de notre unité d'affectation, en passant par les tâches que j'ai effectué ainsi que les intérêts apportés par ce stage.

Table de matière

Chapitre 1 :

I. Présentation Marsa Maroc :

1. Fiche signalétique :

- DENOMINATION :
  MARSA MAROC
- IDENTIFICATION:
  Bd. des Almohades Casablanca Maroc
  ☎ 022.31.71.11 (Tel)
  ☎ 46.426/46.427 (Telex)
  ☎ 022.31.57.21 (Telefax)
- PRESIDENT DU DIRECTOIRE:
  Monsieur Mohammed Abdeljalil
- FORME JURIDIQUE :
  Société Anonyme à directoire et à conseil de surveillance
  (Elle est régie par la loi n°17-95 sur les sociétés anonymes et la loi 15-02 relative aux ports)
- SECTEUR D'ACTIVITE:
  Exploitation et gestion des ports marocains
- CAPITAL :
  733 956 000,00 DH
- EFFECTIF:
  4172 salariés dont 542 cadres supérieurs

2. Historique :

Le développement économique, commercial et industriel ainsi que sa position géographique et l'extension de son territoire, autant de facteurs qui ont poussé le Maroc à renforcer ses infrastructures et s'adapter aux normes modernes du trafic afin de répondre aux besoins du pays et ses partenaires.

A cet égard, le gouvernement a procédé depuis l'indépendance à la création d'un certain nombre d'organisations publiques à caractère commercial et industriel dont

ODEP (Office D'Exploitation des Ports) l'un des meilleurs exemples et qui était crée le 28 décembre 1984. En 2006, il a changé son nom en MARSA MAROC.

Ainsi MARSA MAROC intervient dans les ports de :

- **CASABLANCA**
- **MOHAMMEDIA**
- **JORF LASFER**
- **KENITRA**
- **DAKHLA**
- **NADOR**
- **TANGER**
- **TANTAN**
- **AGADIR**

3. Mission de MARSA MAROC:

Dans le but de développer et de dynamiser le secteur maritime, **MARSA MAROC** a pour mission : Le traitement, dans les meilleures conditions de délai, de coût et de sécurité, de l'ensemble des navires et des marchandises transitant par les ports marocains.

Depuis l'annonce de l'escale d'un navire, jusqu'à la livraison de la marchandise à son propriétaire, divers services seront rendus par **MARSA MAROC** :

- La Gestion du domaine publique , un service d'aide à la navigation
- Le pilotage, le remorquage, le lamanage et l'avitaillement des navires
- La manutention et l'entreposage des marchandises
- Un système d'information permanent.

Outre sa mission d'exploitant portuaire, **MARSA MAROC** est aussi chargé de :

- La maintenance des infrastructures portuaires autres que les ouvrages de protection et les chenaux d'accès.
- La gestion des gares maritimes et leurs annexes.

- La gestion des halles de poissons de Mohammedia, Casablanca, Safi et Agadir
- Par ailleurs, **MARSA MAROC** assure pour le compte de l'état et sous son contrôle, les missions de police portuaire.
- Mission Audit et Contrôle de Gestion (MACG)
- Mission Communication (MC)
- Mission Gestion des Risques (MGR)
- Mission Suivi et Recouvrement des Créances (MSRC)
- Mission Coordination de l'Exploitation des Ports (MCEP)

4. Différentes directions de la société d'exploitation des ports :

4.1) Direction centrale :

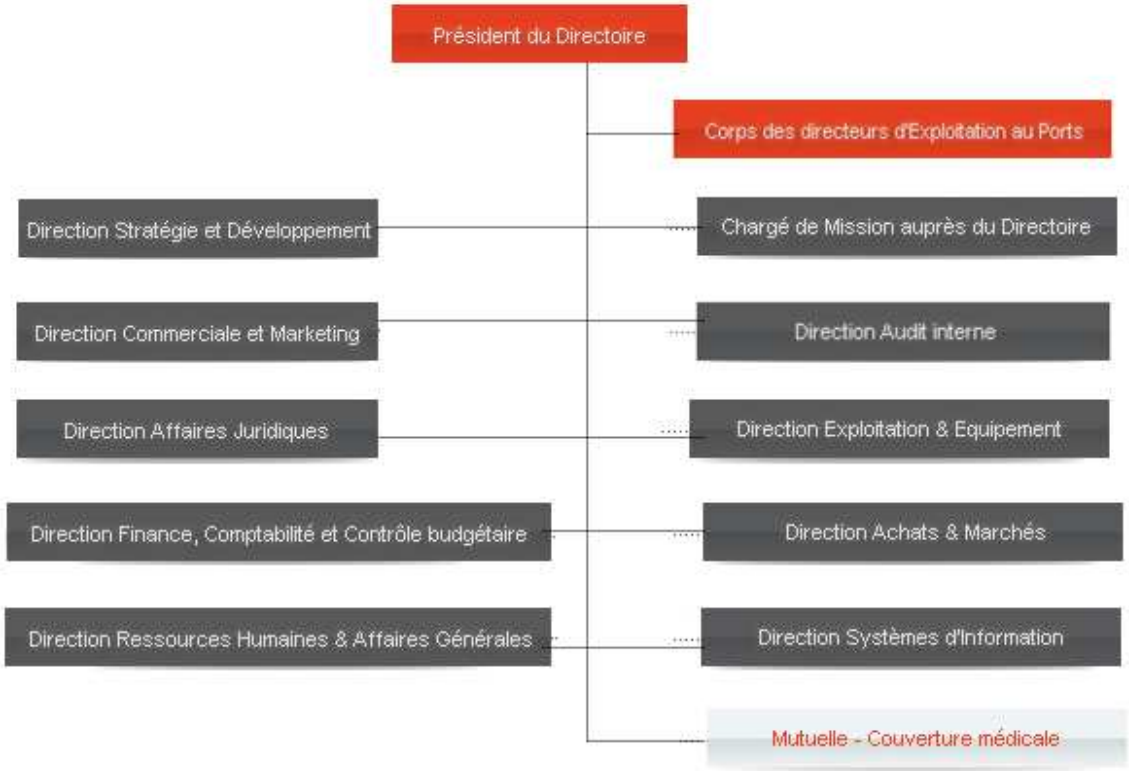
La direction centrale permet de garantir la cohérence de la politique de développement et des orientations stratégiques de l'ensemble des ports, Développer une politique Marketing et commerciale commune, Mettre en place des procédures et modes opératoires pour une meilleure harmonisation et optimisation des pratiques et des règles de l'Exploitation portuaire, Définir les règles communes de gestion en ce qui concerne les fonctions de support et de soutien.

4.2) Neuf directions de l'exploitation aux ports :

Les neuf directions ont pour rôle d'assurer la représentation de la Direction Générale au niveau du port sur l'ensemble de ses missions et attribution.

Chaque direction est responsable de la définition et de la réalisation de sa stratégie suivant les orientations de la direction générale. Elle est, en outre responsable, de l'élaboration de son budget ainsi que son exécution une fois approuvé.

Pour la réalisation de leurs objectifs, chacune des directions d'exploitation est responsable de ses ressources, de son patrimoine (équipements et outillage) et de ses investissements.



5. Présentation de SIPOR :

5.1) Définition et objectifs :

Le Système d'Information portuaire SIPOR automatise l'ensemble des opérations ou procédures au Port qu'elles soient d'ordre commercial, technique, exploitation, financier ou ressources humaines.

Le modèle s'inspire d'une architecture systémique des systèmes d'information. Il est, en effet, composé d'un système opérationnel et d'un système d'aide à la décision complètement intégrés.

La réalisation du système SIPOR a pour objectifs :

- l'automatisation des procédures d'exploitation pour lever les goulots d'étranglement constitués par une charge administrative proportionnelle au volume du trafic portuaire, à sa diversification et à sa complexité.

- la coordination et le contrôle des opérations portuaires pour la communauté portuaire.
- l'automatisation du transfert des documents de transport pour accompagner les flux physiques des marchandises et anticiper la programmation des navires.

5.2)Les sous systèmes principaux de SIPOR :

Le système **SIPOR** est composé de cinq sous-systèmes principaux :

- **TRAFIC** : il traite l'ensemble des procédures d'exploitation relatives au navire et sa marchandise.

- **APIPRO**: il prend en charge la gestion automatisée de la maintenance des équipements et des infrastructures. Il gère aussi les achats et les stocks.

- **SYFCOM**: le système comptable et financier du port.

- **HRACCESS** : il automatise la gestion des ressources humaines et la paie.

- **PREST** : il automatise la gestion commerciale composée d'une base des données client et le système de facturation et de recouvrement.

5.2.1) Présentation du système : TRAFIC

Le système **TRAFIC** automatise l'ensemble des opérations portuaires par la prise en charge informatique de l'escale et de sa marchandise depuis son annonce jusqu'à la liquidation des lots des marchandises qui constituent sa cargaison.

- **La fiche technique du système TRAFIC**

- TRAFIC est un logiciel à 100% développé.
- La programmation de TRAFIC a duré cinq années.
- La mise en exploitation du système **TRAFIC** a été progressive au fur et à mesure de l'avancement des développements.
- Il a été maintenu pendant deux années.
- Le module d'information de gestion est le dernier module en cours de test. Il informatise tous les rapports d'activité en plus des tableaux de bord et les statistiques annuelles.

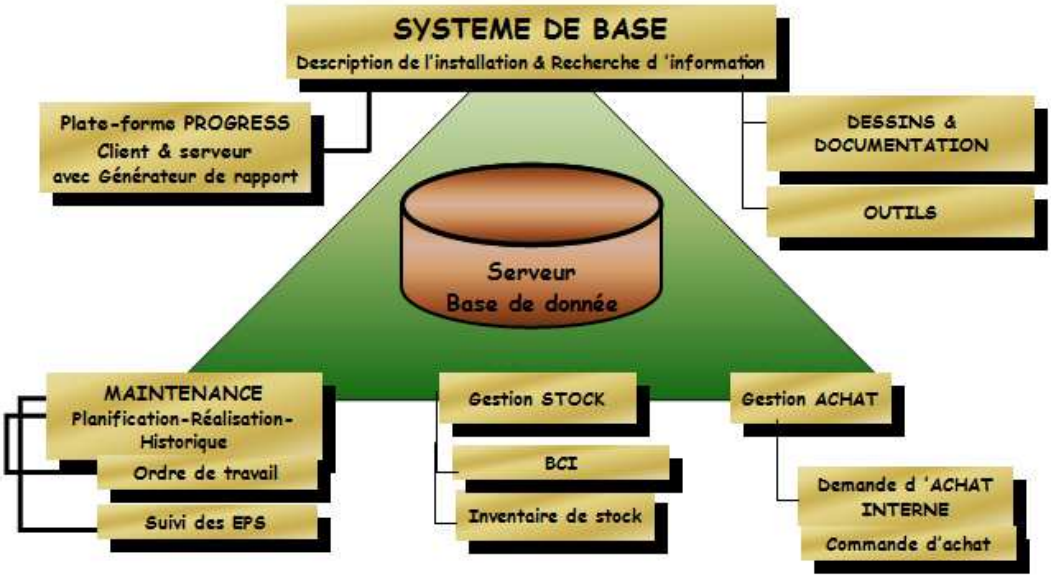
- Tous les types de trafic et toutes les opérations d'exploitation portuaire sont informatisés : tous les visa et les bons de sortie sont informatisés ainsi que l'état différentiel.
- Les modules du système **TRAFIC** installés dans les départements d'Agadir, Safi et Mohammedia sont : Les Prévisions des Escales et les Mouvements des Navires.
- Le nombre des utilisateurs de **TRAFIC** dans les départements dépend des installations futures.
- Le nombre prévisionnel moyen des utilisateurs de **TRAFIC** est estimé à 70 personnes par département.
- Le nombre des utilisateurs cibles de **TRAFIC** est d'environ : 700 personnes directement liés à l'exploitation.

5.2.2)- Présentation du système APIPRO :

La gestion de la maintenance assistée par ordinateur APIPRO doit répondre essentiellement aux objectifs suivants :

- Devenir un moyen des utilisateurs pour atteindre la disponibilité opérationnelle des équipements de la production portuaire (infrastructures, équipements de manutention)
- Aider les utilisateurs pour maîtriser les coûts de la maintenance.
- Apporter une aide au déclenchement automatique des opérations d'entretien préventif et curatif et à leur suivi.
- Optimiser, en le réduisant, le capital immobilisé en pièces de rechange.
- Soulager les techniciens de la maintenance dans les travaux administratifs concernant la connaissance des matériels, leur suivi et la gestion des pièces de rechange.

Structure d' APIPRO



- **La fiche technique du système APIPRO :**

- **APIPRO** est un progiciel acheté. Il a été enrichi par les développements spécifiques à **MARSA MAROC.**
- **APIPRO** est installé à la direction et aux départements. Le nombre des utilisateurs est environ de 447 utilisateurs réparti comme suit: 312 utilisateurs pour l'achat par BCI, 14 utilisateurs pour la gestion des stocks et 121 utilisateurs de la maintenance des équipements.
- Tous les modules du système **APIPRO** sont opérationnels : achat par BCI, gestion des stocks, l'entretien préventif et curatif.
- Les modules du système **APIPRO** installés à la direction générale et dans les départements sont: l'achat par BCI et la gestion des Stocks.
- le nombre des utilisateurs d'**APIPRO** dans les départements est d'environ 20 personnes en moyenne.
- le nombre des utilisateurs actuels d'**APIPRO** est d'environ 593 personnes.

5.2.3) Présentation du système comptable et financier: SYFCOM

L'architecture du système **SYFCOM** incorpore le module 'comptabilité générale et auxiliaires' comme noyau central du produit. Ce noyau permet :

- d'effectuer la tenue de tous les plans de compte : général, auxiliaire et analytique.
- de recevoir tous les flux de gestion provenant des différents sous-systèmes de **SIPOR** et destinés à être comptabilisés.

5 .2.4) La gestion des ressources humaines : HRACCESS :

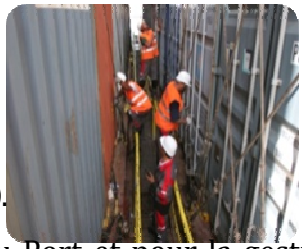
La gestion des ressources humaines forme un système cohérent constitué des principales modules suivants : la paie, la gestion administrative, la gestion de la carrière, la gestion sociale, la gestion de la rémunération et de la masse salariale...

L'objectif du système **HRACCESS** est de pouvoir doter la direction des ressources humaines, les départements et la direction générale, d'un outil susceptible de produire un bilan social complet.

Les fonctionnalités de **HRACCESS** sont présentées sous forme de modules pouvant être mis en œuvre séparément et progressivement pour assurer la maîtrise de la prise en charge par les utilisateurs.

Le système est interfacé aussi avec **TRAFIC** pour la prime de productivité et la disponibilité du personnel. Il est aussi interfacé à **SYFCOM** pour le journal comptable de la paie.

La génération des écritures de la paie sur **SYFCOM** peut être manuelle ou automatique.

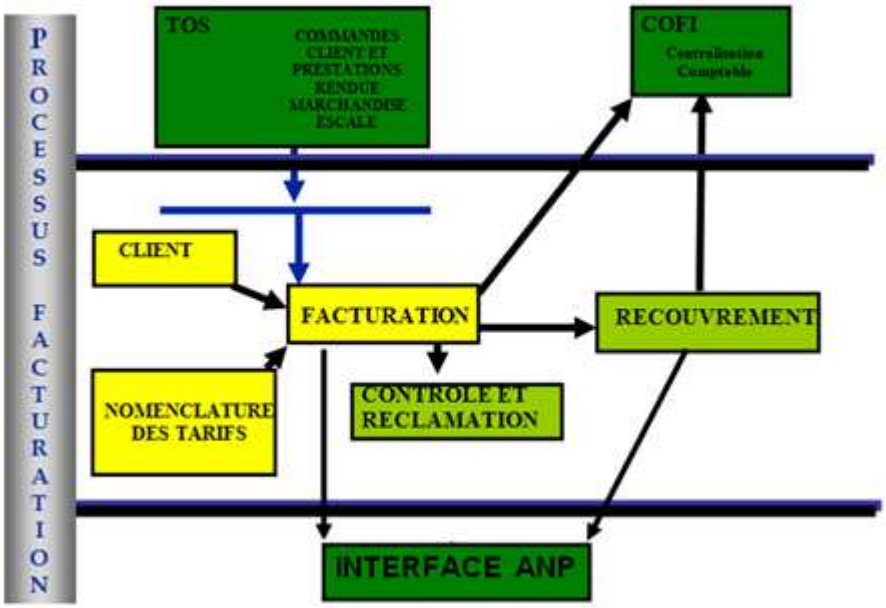


- **La fiche technique du système HRACCESS :**

- **HRACCESS** est un progiciel ERP (***Enterprise Resource Planning)***.
- **HRACCESS** est installé pour la gestion Resource humaine du Port et pour la gestion Centralisé de l'ensemble des ports à la direction générale.
- Les modules du système **HRACCESS** sont opérationnels à la direction générale et dans les Ports.
- le nombre des utilisateurs de **HRACCESS** dans les départements est environ de 10 personnes en moyenne par département.
- le nombre total des utilisateurs prévus de **HRACCESS** est d'environ : 110 personnes

5.2.5) Le système de la gestion commerciale PREST:

La gestion commerciale **PREST** est un sous-système intégré aux autres sous-systèmes de SIPOR.



- **Les fonctions du sous-système PREST :**
    - La nomenclature des tarifs :

Elle est paramétrable ainsi que les éléments qui la constituent:

- Les prestations
- Les catégories des marchandises.
- Les particularités du client : passible de la TVA, son activité, son type (administration...)
- Les réductions
- Les groupes de prestations
- les tarifs.
- Les conventions réalisées avec les clients et les tarifs correspondants.

    - La gestion commerciale :

Permet de gérer :

- La fiche signalétique du client, la gestion des cautions et des avances client.
- Le recouvrement des créances qui supporte les traitements suivants
    - ✓ L'édition des relevés des factures des clients.

- ✓ La relance automatique des clients.
- ✓ Le blocage des prestations aux clients dont la caution résiduelle ne permet plus de rouvrir les prestations commandées.
- ✓ Les encaissements.
- ✓ La réalisation des cautions.
- ✓ La gestion des incidents de paiement.
- ✓ Les réclamations

- Les réclamations qui permettent les traitements suivants:
  - ✓ L'édition des lettres de réclamation.
  - ✓ L'annulation des factures.
  - ✓ La gestion des ristournes.
  - ✓ La gestion des avoirs.

o <u>La facturation :</u>

La facturation concerne :

- Les droits de port sur navire et marchandise.
- Les prestations rendues aux navires et aux marchandises.
- La facturation de l'eau et d'électricité.
- La manutention et le stockage de tous les types de trafic :
  - ✓ Les minerais et les hydrocarbures.
  - ✓ Les agrumes et les primeurs.
  - ✓ Les marchandises conventionnelles.
  - ✓ Les conteneurs.
  - ✓ Les remorques.
  - ✓ Les céréales.
  - ✓ Le transit international par voie terrestre (remorque, véhicules... qui ne sont pas en import ni en export).

Chapitre 2 :

I. Analyse et conception:

1. Définition du besoin :

Lorsqu'une entité technique manifeste un besoin, elle rédige le cahier de prescription spéciale « CPS » et lance un appel d'offre au département d'achat qui le valide ou bien le retourne à l'entité technique pour le modifier.

Ensuite, on détermine deux dates :

- Date d'ouverture des plis .

- Date de publication .

Après avoir rassembler toutes les réponses des fournisseurs (offres techniques) , on passe au jugement selon deux critères :

- Meilleure note : après avoir effectué une étude technique, l'entité concernée attribue une note = (note technique + note financière)/2 et l'attribution du marché revient au fournisseur qui a la meilleure note.

- Moins disant : l'attribution du marché revient au fournisseur dont l'offre est la moins chère.

2. Cahier de charge:

L'objectif du projet est la conception ainsi que la réalisation d'une application qui permet la

gestion des appels d'offres.

Le système doit intégrer les modules suivants:

- Gestion des utilisateurs
- Gestion des appels d'offres
- Gestion des marchés
- Gestion des fournisseurs

Module des utilisateurs:

*Gérer les comptes (utilisateur)

Module des Appels d'Offres:

Ce module permet de gérer le suivi des Appels d'Offres:

* L'objet d'offre

* La date de reception

* Le type

* Les fournisseurs

Module des marchés:

Ce module permet de gérer le suivi des marchés :

* Numéro du marché

* Montant du marché

* Date d'approbation

Module des fournisseurs :

Ce module permet de gérer le suivi des fournisseurs :

* Code

* Nom

* Prénom

* Adresse

- **Règle de gestion:**

RG1 : un client peut avoir comme rôle (administrateur, utilisateur).

RG2 : Un marché résulte d'un seul appel d'offre.

RG3 : Un appel d'offre peut donner naissance à un ou plusieurs marchés.

RG4 : Un fournisseur peut fournir un ou plusieurs appels d'offres.

II. Outils de développement :

1. ADO.NET:

✓ ***Introduction :***

L'**ADO.NET** permet de gérer la base de données, de relier une application à une base de données. Sa nouveauté par rapport à son ancêtre l'**ADO** est la gestion de données dans une application dans un environnement déconnecté. Ce mode, par rapport au mode connecté classique, possède plusieurs avantages et inconvénients (annexe). Ces deux modes sont utilisables avec les fournisseurs de base de données.

✓ **Définition :**

ActiveX Data Object ou **ADO** est une bibliothèque logiciel de Microsoft fournissant une interface d'accès aux données dans l'environnement Windows.

Elle permet aux programmes clients d'accéder aux données, et de les manipuler, dans un fichier ou un serveur de base de données.

ADO.Net est la nouvelle bibliothèque logicielle d'accès aux données. C'est un ensemble de classes, de structures, de types gérant l'accès à des sources de données.

Il y a deux moyens différents d'accéder aux données, chacun d'eux a ses qualités et ses défauts :

- L'accès grâce à un **datareader** qui est très rapide mais ne lit pas les données. La connexion à la base est toujours activée.

- L'accès grâce à un **dataAdapter** qui charge un **dataset**. Cet accès est plus lent que le premier mais permet d'ajouter, de modifier, de supprimer et de lire les données. Il permet aussi de travailler en mode déconnecté de la base donc laisse l'accès pour d'autres applications plus rapidement.

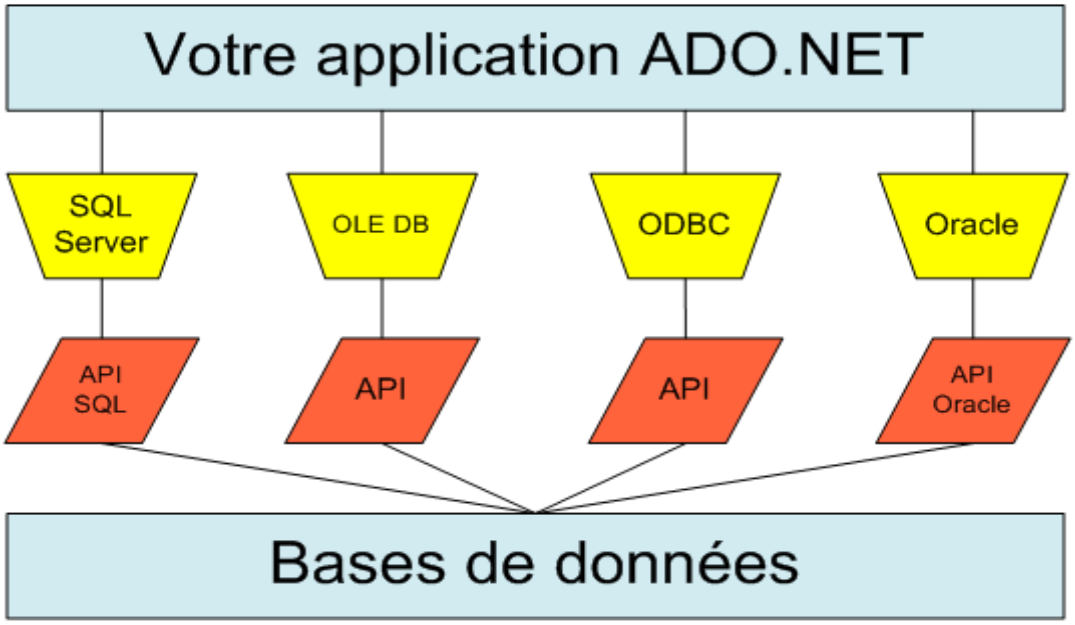
✓ ***Les bases de données***

- **Les fournisseurs de données :**

Chaque fournisseur de données permet la communication avec un type de base de données au travers d'une **API**. Une **API** (*Application Programming Interface*) est l'interface qui permet l'accès du logiciel par un autre. Ces fournisseurs permettent de récupérer et de transférer des modifications

entre l'application et une base de données. Toutes les classes permettant d'utiliser ces fournisseurs se trouvant dans l'espace de nom *System.Data*. Sur le Framework 3.5, il existe quatre types de fournisseurs :

- SQL Server
- OLE DB
- OBDC
- Oracle



Chaque fournisseur est relié à une base de données propre, c'est-à-dire qu'il est compatible à l'**API** de sa base de données. Cependant, les bases de données peuvent implémenter plusieurs **API**) :

Fournisseur	Description
SQL Server	Les classes de ce fournisseur se trouvent dans l'espace de nom *System.Data.SqlClient*, chaque nom de ces classes est préfixé par *Sql*. SQL Server à accès au serveur sans utiliser d'autres couches logicielles le rendant plus performant.
OLE DB	Les classes de ce fournisseur se trouvent dans l'espace de nom *System.Data.OleDb*, chaque nom de ces classes est préfixé par *OleDb*. Ce fournisseur exige l'installation de MDAC (Microsoft Data Access Components). L'avantage de ce fournisseur est qu'il peut dialoguer avec n'importe quelle base de données le temps que le pilote OLE DB est installé dessus, mais par rapport à SQL server, OLE DB utilise une couche logicielle nommée OLE DB ; il requiert donc plus de ressources diminuant par conséquent les performances.
ODBC	Les classes de ce fournisseur se trouvent dans l'espace de nom *System.Data.Odbc*, chaque nom de ces classes est préfixé par *Odbc*. Tout comme l'OLE DB, ODBC exige l'installation de MDAC. Il fonctionne avec le même principe qu'OLE DB mais au lieu d'utiliser une couche logicielle, il utilise le pilote ODBC.
Oracle	Les classes de ce fournisseur se trouvent dans l'espace de nom *System.Data.OracleClient*, chaque nom de ces classes est préfixé par *Oracle*. Il permet simplement de se connecter à une source de données Oracle.

2. Visual Basic .net :

BASIC :

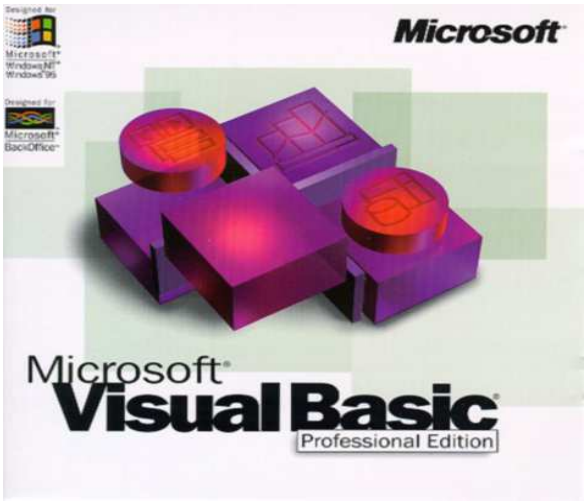
BASIC ou Beginner's All-purpose Symbolic Instruction Code a été conçu pour permettre aux étudiants qui ne travaillaient pas dans des filières scientifiques d'utiliser les ordinateurs.

Les principes de conception du **BASIC** étaient :

- être facile d'utilisation pour les débutants (Beginner) .
- être un langage généraliste (All-purpose) .
- autoriser l'ajout des fonctionnalités pour les experts (tout en gardant le langage simple pour les débutants) .
- fournir des messages d'erreur clairs et conviviaux .

- avoir un délai de réaction faible pour les petits programmes .

VB.net :



Visual Basic.net est dérivé du **BASIC**, le suffixe « .net » spécifie en fait qu'il nécessite le framework **.NET** (une bibliothèque informatique contenant des outils qui facilitent la tache du développeur) de Microsoft afin de pouvoir être exécuté.

VB.net permet le développement rapide d'applications, la création d'interfaces utilisateur graphiques, l'accès aux bases de données en utilisant la technologie **ADO** , ainsi que la création de contrôles ou objets ActiveX

Visual Basic a été conçu pour être facile à apprendre et à utiliser. Le langage permet de créer des applications graphiques de façon simple, mais également de créer des applications véritablement complexes.

Programmer en VB est un mélange de plusieurs tâches, comme disposer visuellement les composants et contrôles sur les formulaires, définir les propriétés et les actions associées à ces composants, et enfin ajouter du code pour ajouter des fonctionnalités.

Comme les attributs et les actions reçoivent des valeurs par défaut, il est possible de créer un programme simple sans que le programmeur ait à écrire de nombreuses lignes de code.

3. Sql server express edition:

- **Définition :**

Le langage **SQL** (Structured Query Language) a été initialement conçu dans les années 1970 par la firme IBM.

Il a été ensuite normalisé et est devenu le standard de tous les SGBDR (système de gestion de base de données relationnelles).

Ce langage permet de masquer aux programmeurs les algorithmes de recherche des données dans des fichiers Physiques eux-mêmes structurés de manière très complexe et différemment selon les SGBDR.

SQL se décompose en trois sous-langages qui s'occupent de :

– La définition des données : création des tables, des contraintes .

– La manipulation des données : sélectionner, insérer, supprimer et modifier .

– Le contrôle des données : intégrité, droits d'accès, verrous et cryptage .

- **Base de données en SQL:**

Une base de données est un objet particulièrement difficile à définir puisqu'il est abordé en pratique selon différents points de vue :

– Pour un utilisateur, une base de données est un espace où il peut enregistrer des informations, les retrouver et les faire traiter automatiquement par un ordinateur (on retrouve là, l'étymologie du mot informatique) .

– Pour un développeur, une base de données est un ensemble de tables, de relations et de procédures écrites en **SQL** .

– Pour un administrateur informatique, une base de données est un ensemble de données à sauvegarder et sécuriser.

➔Nous nous contentons ici du rôle de **développeur**, cela signifie que nous occultons l'administration d'une base de données mais que nous gardons en tête les préoccupations des utilisateurs.

Il y'a deux types de base de données :

-Base de données personnelle:

Dans une base de données personnelle (que l'on manipule dans le logiciel Access de Microsoft par exemple), on retrouve essentiellement un schéma où je suis l'unique concepteur, développeur, fournisseur et analyse des données.

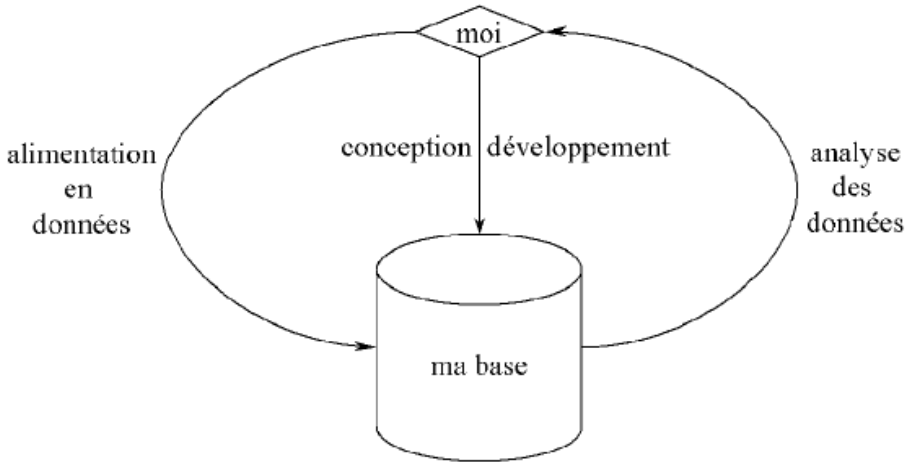


Fig: – Base de données personnelle

-Base de données professionnelles :

Dans un SGBD professionnel (de type **SQL Server**, Oracle, et bien d'autres) , le schéma est fondamentalement différent : les données sont fournies par plusieurs utilisateurs à travers de multiples petites transactions **SQL**. Ces données sont stockées dans une ou Plusieurs bases de production continuellement remises à jour par ces transactions.

Cette partie amont du schéma constitue le système transactionnel. Les données sont en général historisées dans un entrepôt de données dont l'élément constitutif n'est plus la table mais le cube.

Ceci génère de gros transferts entre les deux systèmes mais les informations utiles sont plus proches des quelques utilisateurs qui ont besoin d'analyser les données.

Cette partie aval du schéma constitue le système décisionnel. L'ensemble est géré, dans l'entreprise, par les concepteurs, les développeurs et les administrateurs du service informatique.

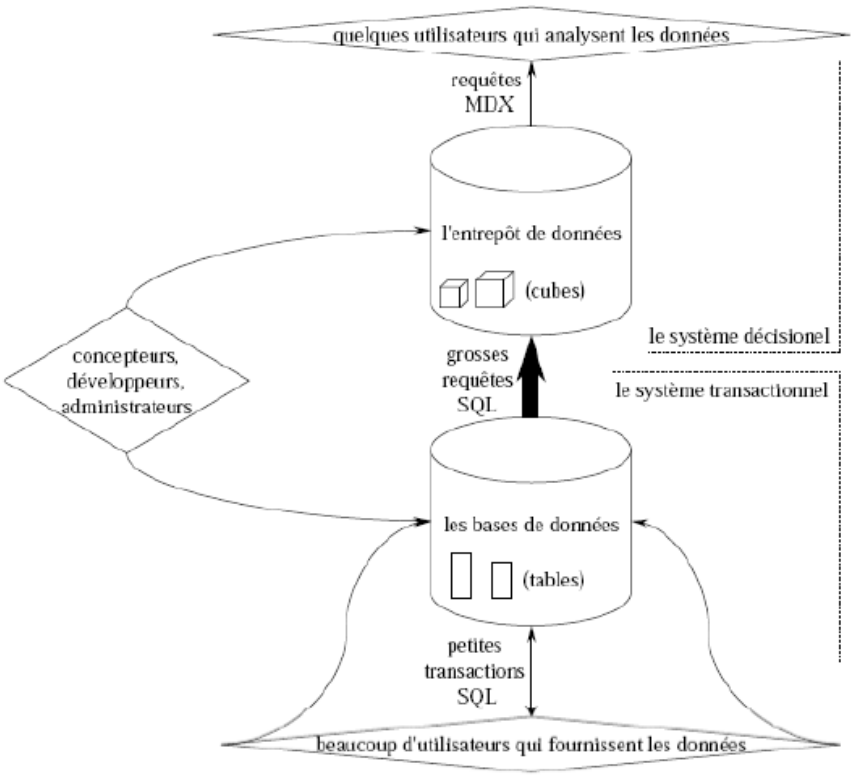
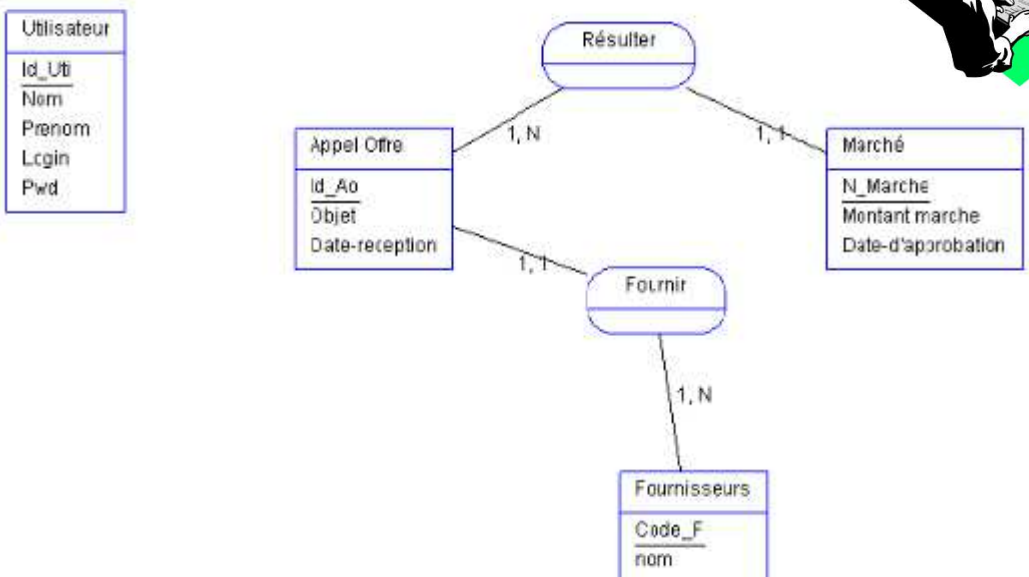


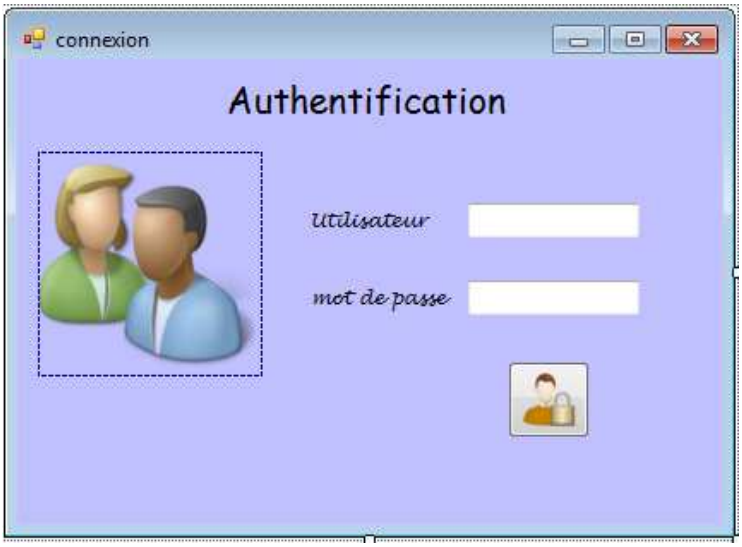
Fig: – Base de données professionnelle

III. Dévellopement de l'application ·

1. MCD: (annexe)



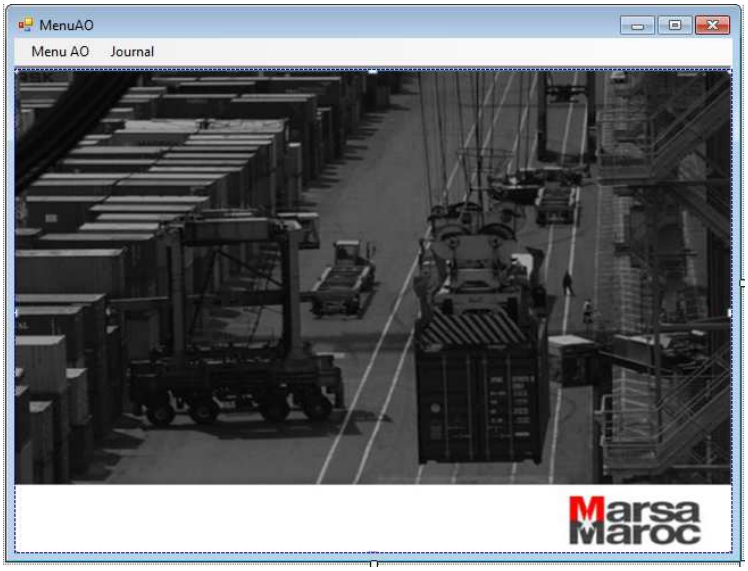
2. Interfaces utilisées :



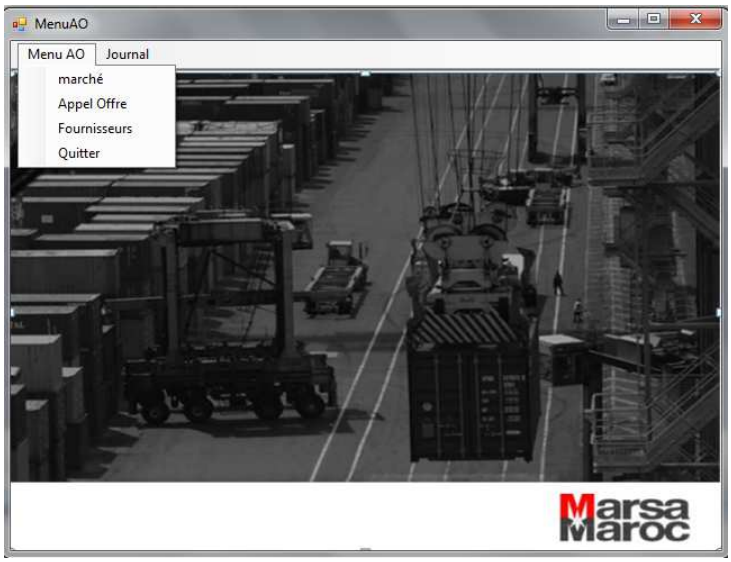
⇨ Cette interface est **l'entrée du projet**, elle permet d'accéder au menu de l'application.

⇨ Le bouton  permet de vérifier si le nom de l'utilisateur et son mot de passe sont corrects pour accéder au menu, sinon un message s'affiche vous signalant que le nom d'utilisateur et/ou le mot de passe est incorrect. (code : annexe)

Une fois le menu apparait : vous avez deux sous-menus : Menu AO et Journal.

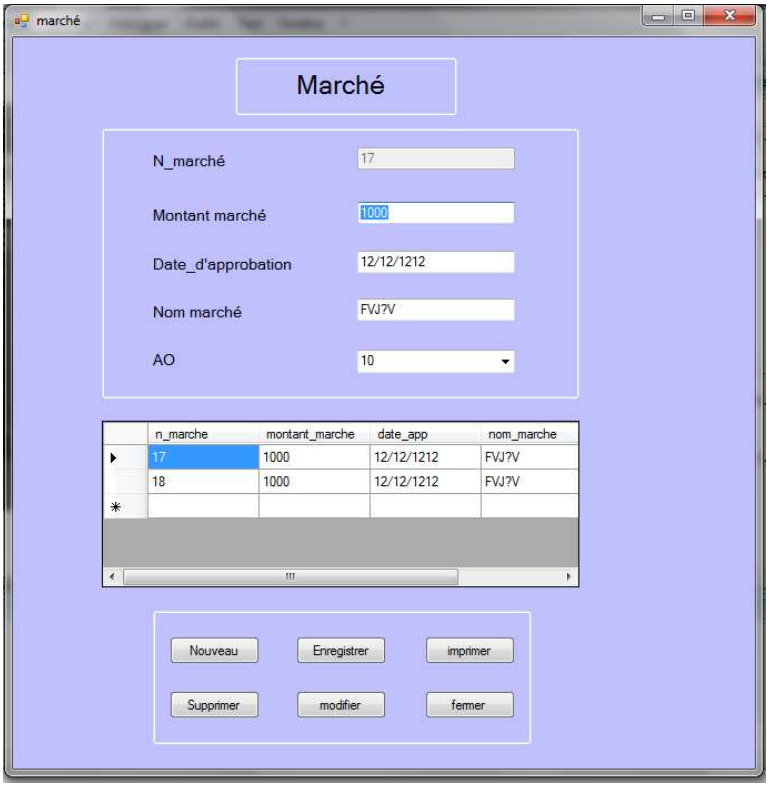


⇨ Si vous Choisissez le « Menu AO » :



Vous avez le choix entre trois tables :

1. <u>Marché :</u>



Cette table comporte les éléments suivants :

⇨ Les champs :

- N_marché qui est la **clé primaire**(annexes) et s'auto-incrémente.
- Montant marché, date_d'approbation et nom marché que nous pouvons ajouter ou modifier manuellement.
- AO (clé secondaire (annexe)) est un comboBox qui contient les Id des appels d'offres disponibles.

⇨ Les boutons :

Nouveau : vide les champs (c'est à dire les TextBOx) pour les remplir et créer un nouveau marché.

Enregistrer : enregistre les modifications après avoir ajouter un nouveau marché.

Modifier : enregistre les modifications après avoir modifier un champ quelconque.

Supprimer : supprime une ligne après l'avoir sélectionnée.

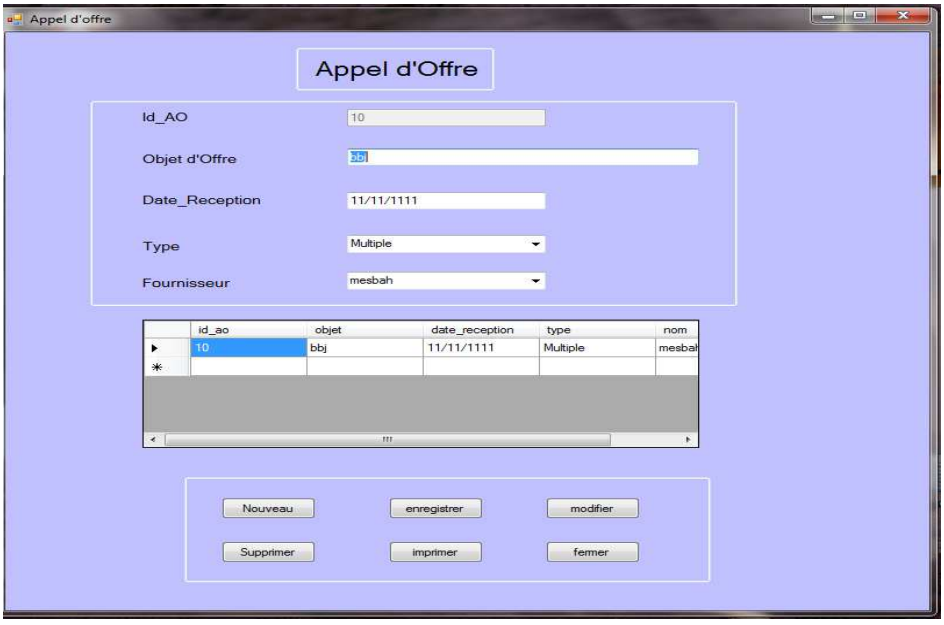
Imprimer : se charge de l'impression.

Fermer : ferme la table uniquement (pas toute l'application).

N.B : le codage de ces boutons est détaillé dans l'annexe.

⇨ Un DataGridView (annexe) où s'affiche les données.

2. <u>Appel d'offre :</u>

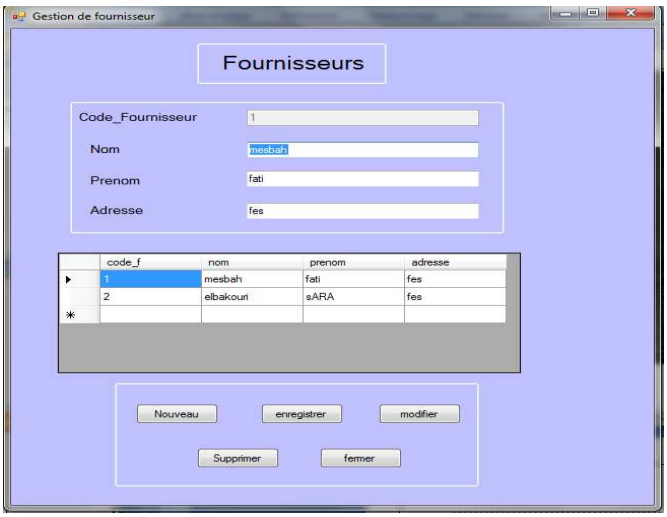


Cette table comporte :

⇨ les champs : Id AO (**clé primaire** et auto-incrémentable), objet d'offre, date_Reception, type (comboBOx qui comporte les deux types disponibles : multiple et unique) et finalement le champ Fournisseur (qui comporte les noms et prénoms des fournisseurs disponibles).

⇨ Le DataGridView et les boutons gardent les mêmes fonctions dans toutes les tables.

3. <u>Fournisseurs :</u>



Cette table comporte :

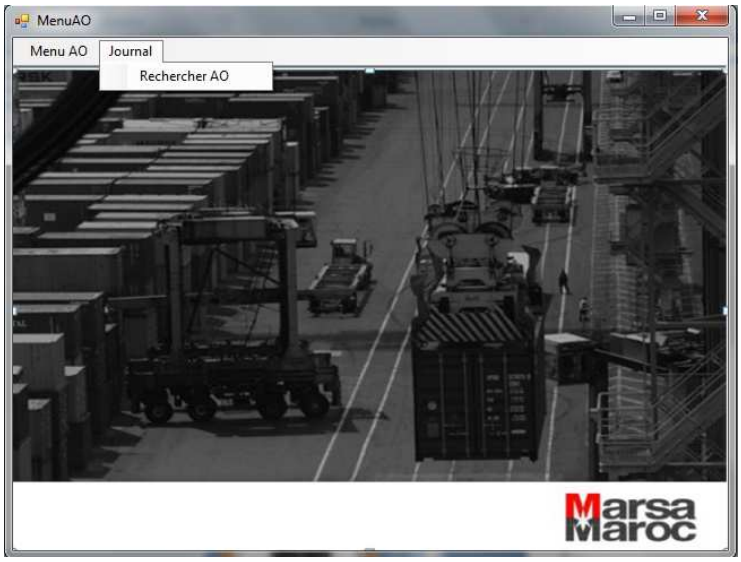
⇨ les champs : code_fournisseur(**clé primaire** et auto-incrémentable), nom, prénom et l'adresse.

⇨ Le DataGridView et les boutons gardent les mêmes fonctions dans toutes les tables.

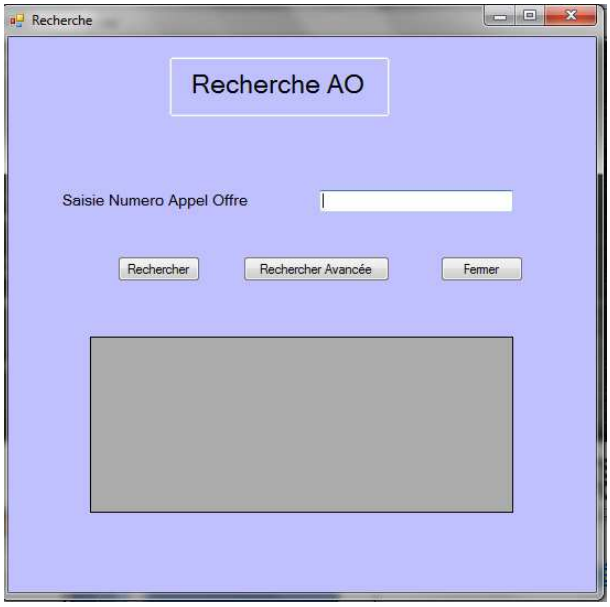
4. <u>Quitter :</u>

Ce n'est pas en effet une table, mais un bouton qui nous permet de quitter toute l'application. (Code Annexe)

➔Si vous Choisissez le « Journal »



⇨ Vous avez la table : Rechercher AO



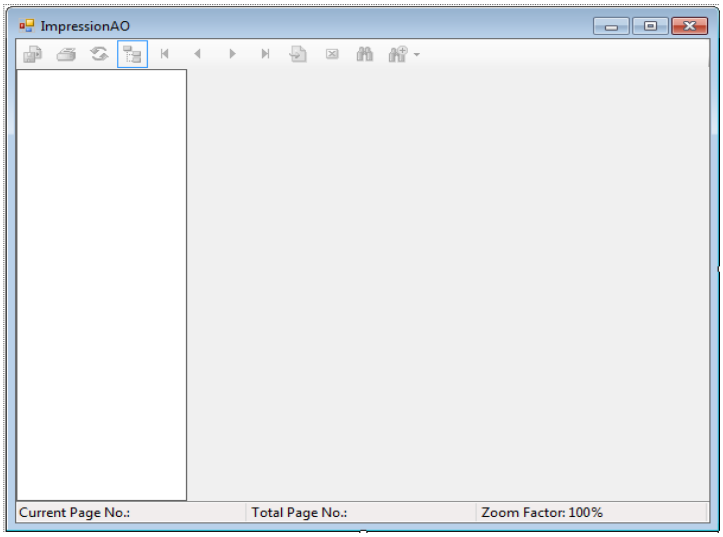
⇨ Cette table sert à **rechercher** un appel d'offre dont je connais uniquement le Numéro. La démarche est la suivante : remplir le champ (saisie numéro Appel Offre), appuyer sur le bouton 'Rechercher' et le résultat de la recherche s'affiche dans le DataGridView.

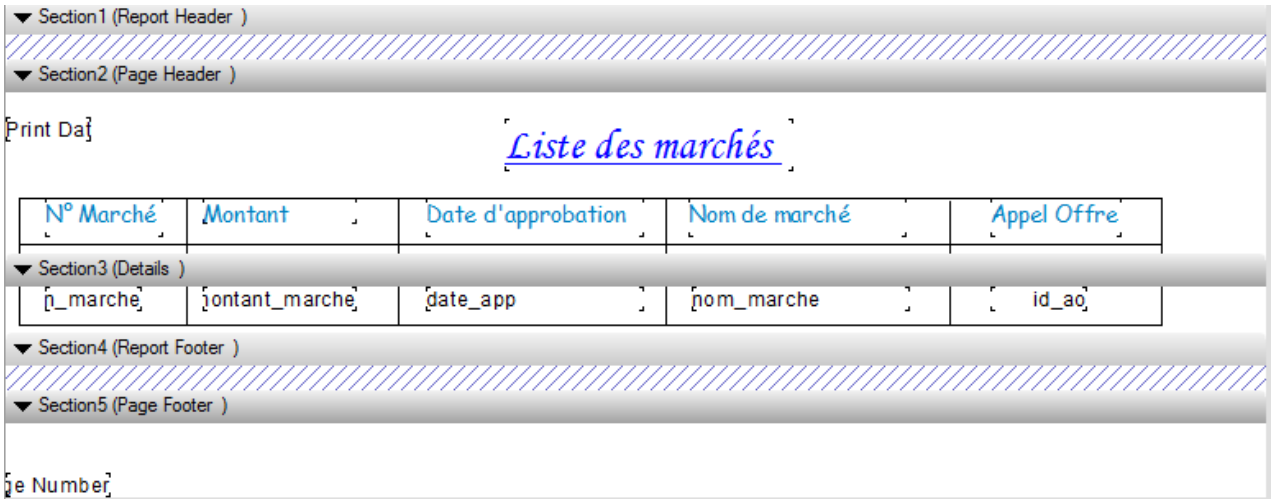
⇨ Le bouton « Recherche Avancée » est utile en cas où l'utilisateur ignore le numéro de l'Appel et connait son Type.

Dans ce cas, en appuyant sur ce bouton, la table suivante apparait ce qui permet à l'utilisateur de rechercher son Appel d'Offre et de garder une copie en l'imprimant grâce au bouton **imprimer.**

L'Impression :

- Pour le créer tout d'abord j' ajoute à l'interface l'événement : CrystalReportViewer.
- Et je crée un CrystalReport , ca représente une facture ou une impresse complétant
  le rôle du CrystalReportViewer.





Exemple du CrystalReport de la table marché

Conclusion

Lors de mon stage de fin d'étude de licence sciences et techniques, j'ai pu développer, en respectant le cahier de charge, une application qui gère les appels d'offre **codée manuellement** au sein de Marsa Maroc.

Pendant les huit semaines de stage , j'ai rencontré plusieurs difficultés pour se familiariser avec les outils de développements vu que durant ma formation de licence j'ai pas pu savoir ce que c'est le développement , ce qui m'a poussé à fournir plusieurs efforts juste pour réunir un bon baguage théorique pour enfin passer à la pratique et réaliser l'application.

Nous tenons à remercier tous ceux qui m'ont facilité la tache.

Annexe :

- **Merise (MCD) :**

MERISE = **M**éthode d'**E**tudes et de **R**éalisation **I**nformatique pour les **S**ystème d'**E**ntreprise

Modèle conceptuel des données (MCD) :

Description des données et des relations en termes de :

-Entité ou individu

-Relation ou association

-Propriété sou attribut

Elle consiste à identifier, à partir d'une description exprimée en langage naturel, les entités et les associations en appliquant les règles suivantes :

- les noms deviennent des entités

- les verbes deviennent des associations

L'exemple suivant qui illustre ce propos est bien trop simple pour que cette méthode conduit à des résultats satisfaisants sur un système d'information de taille plus importante :

Une voiture appartient à un modèle particulier. Les noms sont : « voiture », « modèle ». Le verbe est : « appartient à »

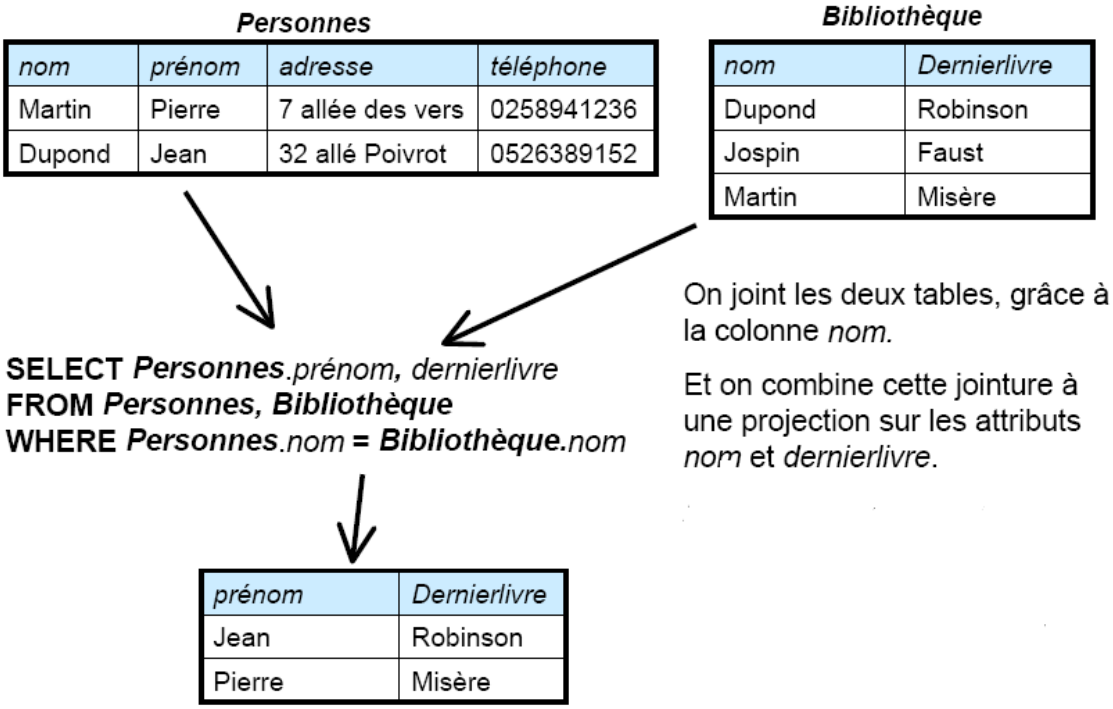
Ce qui donne la modélisation suivante :

- **Jointure:**

Parfois on a besoin d'extraire des champs de plusieurs tables différentes, mais ayant une relation (un champ commun); pour cela on utilise une jointure.

Pour chaque enregistrement de la première table, on affiche en regard les enregistrements de la deuxième table qui ont la même valeur de jointure.

Examples



- **Relation entre les tables:**

Deux tables peuvent être liées et avoir un champ commun présent dans les deux tables.

Sur ce champ commun, il peut exister plusieurs types de relation:

- Relation 1 à N:

➔ Relation un à plusieurs : Un enregistrement unique est lié à plusieurs enregistrements de l'autre table par un champ présent dans les 2 tables.

- Relation 1 à 1:

➔ Relation un à un : Un enregistrement unique est lié à un autre enregistrement unique par un champ présent dans les 2 tables.

- Relation N à M:

➔ Relation plusieurs à plusieurs. Plusieurs enregistrements de la première table peuvent être liés à plusieurs de la seconde table et vice versa.

- **Clé primaire :**

Quand il est nécessaire de **différencier chaque enregistrement de manière unique**, il faut définir un champ comme clé primaire.

Ce champ doit être unique pour chaque enregistrement (il ne doit pas y avoir de doublons: deux enregistrements ne peuvent pas avoir la même clé primaire), et la valeur de la clé primaire ne peut pas être nulle.

La clé primaire ne doit exister que pour faire les liens dans une base de données relationnelle. Exceptionnellement, elle peut être utilisée et montrée à l'utilisateur dans le cas de gestion d'individus ou de personnes pour lesquels aucun attribut ou groupe d'attributs ne permet de discriminer les différentes occurrences d'enregistrements.

- **Clé secondaire :**

Une clé secondaire permet d'accéder à un enregistrement sans devoir connaître la valeur de la clé primaire. Elle est **obligatoire et unique .**

Une clé secondaire sera toujours de type alphanumérique pour garantir son évolutivité

- **Mode connexion / Déconnexion :**

L'ADO.NET permet de séparer les actions d'accès ou de modification d'une base de données. En effet, il est possible de manipuler une base de données sans être connecté à celle-ci, il suffit juste de se connecter pendant un court laps de temps afin de faire une mise à jour. Ceci est possible grâce au DataSet. C'est pourquoi, il existe deux types de fonctionnements :

- Le mode connecté
- Le mode déconnecté

Ci-après, la différence par avantages et inconvénients :

Mode	Avantages	Inconvénients
Connecté	Avec un mode connecté, la connexion est permanente, par conséquence les données sont toujours à jour. De plus il est facile de voir quels sont les utilisateurs connectés et sur quoi ils travaillent. Enfin, la gestion est simple, il y a connexion au début de l'application puis déconnexion à la fin.	L'inconvénient se trouve surtout au niveau des ressources. En effet, tous les utilisateurs ont une connexion permanente avec le serveur. Même si l'utilisateur n'y fait rien la connexion gaspille beaucoup de ressource entrainant aussi des problèmes d'accès au réseau.
Déconnecté	L'avantage est qu'il est possible de brancher un nombre important d'utilisateurs sur le même serveur. En effet, ils se connectent le moins souvent et durant la plus courte durée possible. De plus, avec cet environnement déconnecté, l'application gagne en performance par la disponibilité des ressources pour les connexions.	Les données ne sont pas toujours à jour, ce qui peut aussi entrainer des conflits lors des mises à jour. Il faut aussi penser à prévoir du code pour savoir ce que va faire l'utilisateur en cas de conflits.

Il n'existe pas un mode meilleur que l'autre, tout dépend de l'utilisation que l'on compte en faire.

- **Crystal Reports :**

Crystal Reports est l'outil de création d'états fourni avec Visual Studio .NET.

- **DataGridView :**

Le contrôle **DataGridView** offre un moyen puissant et flexible d'afficher des données sous forme de tableau.

- **Codage :**

 :

```
Public Class connexion

    Private Sub BT_CON_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_CON.Click

        con.Open()
        Dim cmd As New SqlCommand("select * from utilisateur where login='"
& TxtLogin.Text & "' and pwd='" & TxtPwd.Text & "'", con)
        dr = cmd.ExecuteReader
        If dr.HasRows Then
            MenuAO.Show()
        Else
            MsgBox("*¤* Le Login et/ou mot de passe est incorrect !! *¤*",
MsgBoxStyle.Information, "Message d'erreur ..")
            TxtLogin.Clear()
            TxtPwd.Clear()

        End If

        con.Close()
End Sub
```

Table Marché :

```
Imports System.Data.SqlClient
Public Class marché
    Dim NumM As Integer
    'Code de la fonction refresh qui rafraichit la datagrid
    Private Sub Refresh_Form()
        Dim cmdM As New SqlCommand
        Dim daM As New SqlDataAdapter
        Dim dsM As New DataSet
        cmdM = con.CreateCommand
        cmdM.CommandText = "select marche.* from marche,ao where
ao.id_ao=marche.id_ao"
        daM.SelectCommand = cmdM
        daM.Fill(dsM, "marche")
        DataGridView1.DataSource = dsM
        DataGridView1.DataMember = "marche"
        DataGridView1.ReadOnly = True
    End Sub
    Private Sub marché_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        con.Open()
        'Remplissage du Datagridview
        Dim Cmd = New SqlCommand("select marche.* from marche,ao where
ao.id_ao=marche.id_ao", con)
        dr = Cmd.ExecuteReader
        dtMarche.Columns.Clear()
        dtMarche.Clear()
```

```
        dtMarche.Columns.Add("n_marche")
        dtMarche.Columns.Add("montant_marche")
        dtMarche.Columns.Add("date_app")
        dtMarche.Columns.Add("nom_marche")
        dtMarche.Columns.Add("id_ao")
        While dr.Read()
            dtMarche.Rows.Add(dr("n_marche"), dr("montant_marche"),
dr("date_app"), dr("nom_marche"), dr("id_ao"))
        End While
        DataGridView1.DataSource = dtMarche
        'Remplissage des zones de texte de l'interface
        TxtMarche.DataBindings.Add("text", dtMarche, "n_marche")
        TxtMt.DataBindings.Add("text", dtMarche, "montant_marche")
        TxtDate.DataBindings.Add("text", dtMarche, "date_app")
        TxtNomM.DataBindings.Add("text", dtMarche, "nom_marche")
        ComboBox2.DataBindings.Add("text", dtMarche, "id_ao")
        dr.Close()
        'Remplissage du ComboBox AO
        Dim cmdAO As New SqlCommand("select * from ao", con)
        dr = cmdAO.ExecuteReader
        While dr.Read
            ComboBox2.Items.Add(dr("id_ao"))
        End While
        dr.Close()
        con.Close()
    End Sub
'le bouton enregistrer
    Private Sub BT_ENRE_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_ENRE.Click
        con.Open()

        Dim cmdEn As New SqlCommand("insert into marche values(" &
TxtMt.Text & ",'" & TxtDate.Text & "','" & TxtNomM.Text & "'," &
ComboBox2.SelectedItem & ")", con)
        If cmdEn.ExecuteNonQuery() Then
            MsgBox("L'ajout est réussit .", MsgBoxStyle.Information,
"Enregistrer")
        Else
            MsgBox("L'ajout n'est pas réussit .", MsgBoxStyle.Information,
"Enregistrer")
        End If
        Refresh_Form()
        con.Close()
    End Sub
           'Le bouton modifier
    Private Sub BT_MODI_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_MODI.Click
        con.Open()
        Dim cmdM As New SqlCommand("update marche set montant_marche=" &
TxtMt.Text & ", date_app='" & TxtDate.Text & "', nom_marche='" &
TxtNomM.Text & "', id_ao=" & ComboBox2.SelectedItem & " where n_marche=" &
TxtMarche.Text, con)
        If MsgBox("Voulez-vous modifier le marché " & TxtMarche.Text, 36,
"Modification ") = MsgBoxResult.Yes Then
            If cmdM.ExecuteNonQuery() Then
                MsgBox("La modification est réussite .",
MsgBoxStyle.Information, "Modifier")
            Else
                MsgBox("La modification n'est pas réussite .",
MsgBoxStyle.Information, "modifier")
```

```
            End If
        End If
        Refresh_Form()
        con.Close()
    End Sub

    Private Sub BT_SUPR_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_SUPR.Click
        con.Open()
        Dim cmdS As New SqlCommand("delete from marche where n_marche=" &
TxtMarche.Text, con)
        If MsgBox("Voulez-vous supprimer le marché " & TxtMarche.Text, 36,
"Suppression ") = MsgBoxResult.Yes Then
            If cmdS.ExecuteNonQuery() Then
                MsgBox("La suppression est réussite .",
MsgBoxStyle.Information, "Supprimer")
            Else
                MsgBox("La suppression n'est pas réussite .",
MsgBoxStyle.Information, "Supprimer")
            End If
        End If
        Refresh_Form()
        con.Close()
    End Sub
    ' code du bouton fermer
    Private Sub BT_FERM_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_FERM.Click
        Me.Close()
    End Sub
    ' code du bouton nouveau
    Private Sub BT_NOUV_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_NOUV.Click
        TxtDate.Clear()
        TxtMarche.Clear()
        TxtMt.Clear()
        TxtNomM.Clear()
        ComboBox2.Text = Nothing
    End Sub

    Private Sub BT_IMPR_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_IMPR.Click
        ImpressionMarché.Show()
    End Sub
    ' Code du DataGridView1
    Private Sub DataGridView1_CellMouseClick(ByVal sender As Object, ByVal
e As System.Windows.Forms.DataGridViewCellMouseEventArgs) Handles
DataGridView1.CellMouseClick
        NumM = DataGridView1.Rows(e.RowIndex).Cells(0).Value
        con.Open()
        Dim cmdAff As New SqlCommand("select marche.* from marche,ao where
ao.id_ao=marche.id_ao and n_marche=" & NumM, con)
        dr = cmdAff.ExecuteReader
        While dr.Read
            TxtMarche.Text = dr(0)
            TxtMt.Text = dr(1)
            TxtDate.Text = dr(2)
            TxtNomM.Text = dr(3)
            ComboBox2.Text = dr(4)
        End While
        dr.Close()
```

```
        con.Close()
    End Sub

    Private Sub TxtMarche_TextChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles TxtMarche.TextChanged

    End Sub
      End Class
```

Table Appel d'offre :

```
Imports System.Data.SqlClient
Public Class AO
    Dim IdAO As Integer

    'Code to Refresh the datagridview
    Private Sub Refresh_Form()
        Dim cmdAO As New SqlCommand
        Dim daAO As New SqlDataAdapter
        Dim dsAO As New DataSet
        cmdAO = con.CreateCommand
        cmdAO.CommandText = "select ao.*,fournisseur.nom from
ao,fournisseur where fournisseur.code_f=ao.code_f"
        daAO.SelectCommand = cmdAO
        daAO.Fill(dsAO, "ao")
        DataGridView1.DataSource = dsAO
        DataGridView1.DataMember = "ao"
        DataGridView1.ReadOnly = True
    End Sub

    Private Sub Form3_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        con.Open()
        'Remplissage datagridview
        Dim Cmd = New SqlCommand("select ao.*,fournisseur.nom from
ao,fournisseur where fournisseur.code_f=ao.code_f", con)
        dr = Cmd.ExecuteReader
        dtAO.Columns.Clear()
        dtAO.Clear()
        dtAO.Columns.Add("id_ao")
        dtAO.Columns.Add("objet")
        dtAO.Columns.Add("date_reception")
        dtAO.Columns.Add("type")
        dtAO.Columns.Add("nom")
        While dr.Read()
            dtAO.Rows.Add(dr("id_ao"), dr("objet"), dr("date_reception"),
dr("type"), dr("nom"))
        End While
        DataGridView1.DataSource = dtAO
        dr.Close()
        'Remplissage des zones de texte

        TxtAO.DataBindings.Add("text", dtAO, "id_ao")
        TxtObjet.DataBindings.Add("text", dtAO, "objet")
        TxtDate.DataBindings.Add("text", dtAO, "date_reception")
        ComboType.DataBindings.Add("text", dtAO, "type")
        ComboF.DataBindings.Add("text", dtAO, "nom")

        'Remplissage de Fournisseur
```

```
        Dim cmdF As New SqlCommand("select * from fournisseur", con)
        dr = cmdF.ExecuteReader
        While dr.Read
            ComboF.Items.Add(dr("nom") + " " + dr("prenom"))
        End While
        dr.Close()
        con.Close()
    End Sub

    Private Sub BT_ENR_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_ENR.Click
        con.Open()
        Dim cmdEn As New SqlCommand("insert into ao values('" &
TxtObjet.Text & "','" & TxtDate.Text & "','" & ComboType.SelectedItem &
"'," & (ComboF.SelectedIndex + 1) & ")", con)
        If cmdEn.ExecuteNonQuery() Then
            MsgBox("L'ajout est réussit .", MsgBoxStyle.Information,
"Enregistrer")
        Else
            MsgBox("L'ajout n'est pas réussit .", MsgBoxStyle.Information,
"Enregistrer")
        End If
        Refresh_Form()
        con.Close()
    End Sub

    Private Sub BT_FER_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_FER.Click
        Me.Close()
    End Sub

    Private Sub BT_NOU_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_NOU.Click
        TxtAO.Clear()
        TxtDate.Text = ""
        TxtObjet.Clear()
        ComboF.Text = Nothing
        ComboType.Text = Nothing
    End Sub

    Private Sub BT_SUP_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_SUP.Click
        con.Open()
        Dim cmdS As New SqlCommand("delete from ao where id_ao=" &
TxtAO.Text, con)
        If MsgBox("Voulez-vous supprimer l'appel d'offre " & TxtAO.Text,
36, "Suppression ") = MsgBoxResult.Yes Then
            If cmdS.ExecuteNonQuery() Then
                MsgBox("La suppression est réussite .",
MsgBoxStyle.Information, "Supprimer")
                TxtAO.Clear()
                TxtDate.Clear()
                TxtObjet.Clear()
                ComboType.Text = ""
                ComboF.Text = ""
            Else
                MsgBox("La suppression n'est pas réussite .",
MsgBoxStyle.Information, "Supprimer")
            End If

        End If
```

```
        Refresh_Form()
        con.Close()
    End Sub

    Private Sub BT_MOD_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_MOD.Click
        con.Open()
        Dim cmdM As New SqlCommand("update ao set objet='" & TxtObjet.Text
& "', date_reception='" & TxtDate.Text & "', type='" &
ComboType.SelectedItem & "' where id_ao=" & TxtAO.Text, con)
        cmdM.ExecuteNonQuery()
        If MsgBox("Voulez-vous modifier l'appel d'offre " & TxtAO.Text, 36,
"Modification ") = MsgBoxResult.Yes Then
            If cmdM.ExecuteNonQuery() Then
                MsgBox("La modification est réussite .",
MsgBoxStyle.Information, "Modifier")
            Else
                MsgBox("La modification n'est pas réussite .",
MsgBoxStyle.Information, "modifier")
            End If
        End If
        Refresh_Form()
        con.Close()
    End Sub

    Private Sub BT_IMP_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_IMP.Click
        ImpressionAO.Show()
    End Sub

    Private Sub DataGridView1_CellMouseClick(ByVal sender As Object, ByVal
e As System.Windows.Forms.DataGridViewCellMouseEventArgs) Handles
DataGridView1.CellMouseClick
        IdAO = DataGridView1.Rows(e.RowIndex).Cells(0).Value
        con.Open()
        Dim cmdAff As New SqlCommand("select ao.*,fournisseur.nom from
ao,fournisseur where fournisseur.code_f=ao.code_f and id_ao=" & IdAO, con)
        dr = cmdAff.ExecuteReader
        While dr.Read
            TxtAO.Text = dr("id_ao")
            TxtObjet.Text = dr("objet")
            TxtDate.Text = dr("date_reception")
            ComboType.Text = dr("type")
            ComboF.Text = dr("nom")
        End While
        dr.Close()
        con.Close()
    End Sub

                End Class
```

Table Fournisseurs :

```
Dim idF As Integer
    'Code to Refresh
    Private Sub Refresh_Form()
        Dim cmdF As New SqlCommand
        Dim daF As New SqlDataAdapter
        Dim dsf As New DataSet
        cmdF = con.CreateCommand
        cmdF.CommandText = "select * from fournisseur"
```

```
        daF.SelectCommand = cmdF
        daF.Fill(dsf, "fournisseur")
        DataGridView1.DataSource = dsF
        DataGridView1.DataMember = "fournisseur"
        DataGridView1.ReadOnly = True
    End Sub
    Private Sub fournisseurs_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        con.Open()
        'Rempissage de dataGridView
        Dim Cmd = New SqlCommand("select * from fournisseur", con)
        dr = Cmd.ExecuteReader
        dtF.Columns.Clear()
        dtF.Clear()
        dtF.Columns.Add("code_f")
        dtF.Columns.Add("nom")
        dtF.Columns.Add("prenom")
        dtF.Columns.Add("adresse")
        While dr.Read()
            dtF.Rows.Add(dr("code_f"), dr("nom"), dr("prenom"),
dr("adresse"))
        End While
        DataGridView1.DataSource = dtF
        'remplissage de Zone de texte
        TxtCodeF.DataBindings.Add("text", dtF, "code_f")
        TxtNom.DataBindings.Add("text", dtF, "nom")
        TxtPrenom.DataBindings.Add("text", dtF, "prenom")
        TxtAdresse.DataBindings.Add("text", dtF, "adresse")
        dr.Close()
        con.Close()
    End Sub

    Private Sub BT_NOU_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_NOU.Click
        TxtAdresse.Clear()
        TxtCodeF.Clear()
        TxtNom.Clear()
        TxtPrenom.Clear()
    End Sub

    Private Sub BT_FER_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_FER.Click
        Me.Close()
    End Sub

    Private Sub BT_ENR_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_ENR.Click
        con.Open()
        Dim cmdEn As New SqlCommand("insert into fournisseur values('" &
TxtNom.Text & "','" & TxtPrenom.Text & "','" & TxtAdresse.Text & "')", con)
        If cmdEn.ExecuteNonQuery() Then
            MsgBox("L'ajout est réussit .", MsgBoxStyle.Information,
"Enregistrer")
        Else
            MsgBox("L'ajout n'est pas réussit .", MsgBoxStyle.Information,
"Enregistrer")
        End If
        Refresh_Form()
        con.Close()
    End Sub
```

```
    Private Sub BT_MOD_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_MOD.Click
        con.Open()
        Dim cmdM As New SqlCommand("update fournisseur set nom='" &
TxtNom.Text & "', prenom='" & TxtPrenom.Text & "', adresse='" &
TxtAdresse.Text & "' where code_f=" & TxtCodeF.Text, con)
        If MsgBox("Voulez-vous modifier le fournisseur " & TxtNom.Text & "
" & TxtPrenom.Text, 36, "Modification ") = MsgBoxResult.Yes Then
            If cmdM.ExecuteNonQuery() Then
                MsgBox("La modification est réussite .",
MsgBoxStyle.Information, "Modifier")
            Else
                MsgBox("La modification n'est pas réussite .",
MsgBoxStyle.Information, "modifier")
            End If
        End If
        Refresh_Form()
        con.Close()
    End Sub

    Private Sub BT_SUP_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_SUP.Click
        con.Open()
        Dim cmdS As New SqlCommand("delete from fournisseur where code_f="
& TxtCodeF.Text, con)
        If MsgBox("Voulez-vous supprimer le fournisseur " & TxtNom.Text & "
" & TxtPrenom.Text, 36, "Suppression ") = MsgBoxResult.Yes Then
            If cmdS.ExecuteNonQuery() Then
                MsgBox("La suppression est réussite .",
MsgBoxStyle.Information, "Supprimer")
                TxtCodeF.Clear()
                TxtNom.Clear()
                TxtPrenom.Clear()
                TxtAdresse.Clear()
            Else
                MsgBox("La suppression n'est pas réussite .",
MsgBoxStyle.Information, "Supprimer")
            End If
        End If
        Refresh_Form()
        con.Close()
    End Sub

    Private Sub DataGridView1_CellMouseClick(ByVal sender As Object, ByVal
e As System.Windows.Forms.DataGridViewCellMouseEventArgs) Handles
DataGridView1.CellMouseClick
        IdF = DataGridView1.Rows(e.RowIndex).Cells(0).Value
        con.Open()
        Dim cmdAff As New SqlCommand("select * from fournisseur where
code_f=" & idF, con)
        dr = cmdAff.ExecuteReader
        While dr.Read
            TxtCodeF.Text = dr(0)
            TxtNom.Text = dr(1)
            TxtPrenom.Text = dr(2)
            TxtAdresse.Text = dr(3)
        End While
        dr.Close()
        con.Close()
    End Sub
```

```
End Class
```

Table Recherche AO :

```
Imports System.Data.SqlClient
Public Class RechercheAO

    Private Sub LBL_REC_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles LBL_REC.Click
        dtAO.Clear()
    End Sub

    Private Sub BT_REC_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_REC.Click
        con.Open()
        Dim cmdAO As New SqlCommand("select * from ao where id_ao=" &
TextBox1.Text, con)
        dr = cmdAO.ExecuteReader
        dtAO.Columns.Clear()
        dtAO.Clear()
        dtAO.Columns.Add("IDAO")
        dtAO.Columns.Add("objet")
        dtAO.Columns.Add("date_reception")
        dtAO.Columns.Add("type")

        While dr.Read()
            dtAO.Clear()
            dtAO.Rows.Add(dr("id_ao"), dr("objet"), dr("date_reception"),
dr("type"))
        End While
        DataGridView1.DataSource = dtAO
        TextBox1.Clear()
        con.Close()
    End Sub

    Private Sub BT_RECH_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_RECH.Click
        RechercheAvancee.Show()
    End Sub

    Private Sub BT_FER_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_FER.Click
        Me.Close()
    End Sub

    Private Sub RechercheAO_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

    End Sub
      End Class
```

Table Recherche Avancée :

```
Imports System.Data.SqlClient
Public Class RechercheAvancee

    Private Sub RechercheAvancee_Load(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles MyBase.Load
        'Remplissage de TypeAO
        con.Open()
```

```
        Dim cmdAO As New SqlCommand("select * from ao", con)
        dr = cmdAO.ExecuteReader
        While dr.Read
            ComboType.Items.Add(dr("type"))
        End While
        dr.Close()
        con.Close()
    End Sub

    Private Sub BT_REC_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_REC.Click
        con.Open()

        Dim cmdT As New SqlCommand("select * from ao where type='" &
ComboType.SelectedItem & "'", con)
        Dim dtType As New DataTable
        dtType.Columns.Add("id_ao")
        dtType.Columns.Add("objet")
        dtType.Columns.Add("date_reception")
        dtType.Columns.Add("type")
        dr = cmdT.ExecuteReader
        While dr.Read
            dtType.Rows.Add(dr("id_ao"), dr("objet"), dr("date_reception"),
dr("type"))
        End While
        DataGridView1.DataSource = dtType
        dr.Close()
        con.Close()
    End Sub

    Private Sub BT_FER_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_FER.Click
        Me.Close()
    End Sub

    Private Sub BT_RECH_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BT_RECH.Click
        ImpressionType.Show()
    End Sub
End Class
```

Table MenuAO :

```
Imports System.Windows.Forms

Public Class MenuAO

    Private Sub ShowNewForm(ByVal sender As Object, ByVal e As EventArgs)
        ' Créez une nouvelle instance du formulaire enfant.
        Dim ChildForm As New System.Windows.Forms.Form
        ' Configurez-la autant qu'enfant de ce formulaire MDI avant de
l'afficher.
        ChildForm.MdiParent = Me

        m_ChildFormNumber += 1
        ChildForm.Text = "Fenêtre " & m_ChildFormNumber

        ChildForm.Show()
    End Sub

    Private Sub OpenFile(ByVal sender As Object, ByVal e As EventArgs)
```

```
        Dim OpenFileDialog As New OpenFileDialog
        OpenFileDialog.InitialDirectory =
My.Computer.FileSystem.SpecialDirectories.MyDocuments
        OpenFileDialog.Filter = "Fichiers texte (*.txt)|*.txt|Tous les
fichiers (*.*)|*.*"
        If (OpenFileDialog.ShowDialog(Me) =
System.Windows.Forms.DialogResult.OK) Then
            Dim FileName As String = OpenFileDialog.FileName
            ' TODO : ajoutez le code ici pour ouvrir le fichier.
        End If
    End Sub

    Private Sub SaveAsToolStripMenuItem_Click(ByVal sender As Object, ByVal
e As EventArgs)
        Dim SaveFileDialog As New SaveFileDialog
        SaveFileDialog.InitialDirectory =
My.Computer.FileSystem.SpecialDirectories.MyDocuments
        SaveFileDialog.Filter = "Fichiers texte (*.txt)|*.txt|Tous les
fichiers (*.*)|*.*"

        If (SaveFileDialog.ShowDialog(Me) =
System.Windows.Forms.DialogResult.OK) Then
            Dim FileName As String = SaveFileDialog.FileName
            ' TODO : ajoutez le code ici pour enregistrer le contenu actuel
du formulaire dans un fichier.
        End If
    End Sub


    Private Sub ExitToolsStripMenuItem_Click(ByVal sender As Object, ByVal
e As EventArgs)
        Me.Close()
    End Sub

    Private Sub CutToolStripMenuItem_Click(ByVal sender As Object, ByVal e
As EventArgs)
        ' Utilisez My.Computer.Clipboard pour insérer les images ou le
texte sélectionné dans le Presse-papiers
    End Sub

    Private Sub CopyToolStripMenuItem_Click(ByVal sender As Object, ByVal e
As EventArgs)
        ' Utilisez My.Computer.Clipboard pour insérer les images ou le
texte sélectionné dans le Presse-papiers
    End Sub

    Private Sub PasteToolStripMenuItem_Click(ByVal sender As Object, ByVal
e As EventArgs)
        'Utilisez My.Computer.Clipboard.GetText() ou
My.Computer.Clipboard.GetData pour extraire les informations du Presse-
papiers.
    End Sub

    Private Sub CascadeToolStripMenuItem_Click(ByVal sender As Object,
ByVal e As EventArgs)
        Me.LayoutMdi(MdiLayout.Cascade)
    End Sub

    Private Sub TileVerticalToolStripMenuItem_Click(ByVal sender As Object,
ByVal e As EventArgs)
        Me.LayoutMdi(MdiLayout.TileVertical)
```

```
    End Sub

    Private Sub TileHorizontalToolStripMenuItem_Click(ByVal sender As
Object, ByVal e As EventArgs)
        Me.LayoutMdi(MdiLayout.TileHorizontal)
    End Sub

    Private Sub ArrangeIconsToolStripMenuItem_Click(ByVal sender As Object,
ByVal e As EventArgs)
        Me.LayoutMdi(MdiLayout.ArrangeIcons)
    End Sub

    Private Sub CloseAllToolStripMenuItem_Click(ByVal sender As Object,
ByVal e As EventArgs)
        ' Fermez tous les formulaires enfants du parent.
        For Each ChildForm As Form In Me.MdiChildren
            ChildForm.Close()
        Next
    End Sub

    Private m_ChildFormNumber As Integer

    Private Sub MToolStripMenuItem_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MToolStripMenuItem.Click

    End Sub

    Private Sub QuitterToolStripMenuItem1_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
QuitterToolStripMenuItem1.Click
        If MsgBox("Souhaitez-vous vraiment quitter  ?", 36, "Quitter") =
MsgBoxResult.Yes Then
            End
        End If
    End Sub

    Private Sub EToolStripMenuItem_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles EToolStripMenuItem.Click
        marché.Show()
    End Sub

    Private Sub MToolStripMenuItem1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MToolStripMenuItem1.Click
        AO.Show()
    End Sub

    Private Sub FournisseursToolStripMenuItem1_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
FournisseursToolStripMenuItem1.Click
        fournisseurs.Show()
    End Sub

    Private Sub MenuAO_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    End Sub
    Private Sub RechercherAOToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
RechercherAOToolStripMenuItem.Click
        RechercheAO.Show()
    End Sub
End Class
```

Impression AO:

```
Imports System.Data.SqlClient

Public Class ImpressionAO
    Dim crAO As New CrystalReport3

    Private Sub ImpressionAO_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        con.Open()
        Dim cmdM As New SqlCommand("select ao.*,fournisseur.nom from
ao,fournisseur where ao.code_f=fournisseur.code_f and ao.id_ao=" &
AO.TxtAO.Text, con)
        dr = cmdM.ExecuteReader
        Dim dt As New DataTable
        dt.Columns.Add("id_ao")
        dt.Columns.Add("objet")
        dt.Columns.Add("date_reception")
        dt.Columns.Add("type")
        'dt.Columns.Add("nom")
        ''dt.Columns.Add("prenom")
        While dr.Read()
            dt.Rows.Add(dr("id_ao"), dr("objet"), dr("date_reception"),
dr("type"))
        End While
        crAO.SetDataSource(dt)
        CrystalReportViewer1.ReportSource = crAO

        con.Close()
    End Sub
End Class
```