

TÉCNICO

SAP ABAP

PROGRAMACIÓN AVANZADA DE
APLICACIONES COMERCIALES



Tabla de Contenido:



1. ¿Qué es un ERP?:	3
2. ¿Qué es SAP?:	4
2.1. Versiones de SAP:	4
3. Arquitectura de un Sistema SAP:	5
3.1. SAP GUI:	8
4. Principales módulos SAP:	8
5. Conceptos Básicos:	9
5.1. Estructura Organizativa:	10
6. ABAP:	12
6.1. Transacciones:	13
6.2. Diccionario de datos:	14
7. ¿Cuál es la estructura de un programa?:	20
8. Palabras Clave para programar:	28

1. ¿Qué es un ERP?:

Un ERP (Enterprise Resource Planning → Planificación de recursos empresariales) es un conjunto de programas integrados que ayuda a resolver las tareas de las diferentes áreas de una empresa, como puede ser la producción y la logística, finanzas y contabilidad, ventas y recursos humanos, etc.

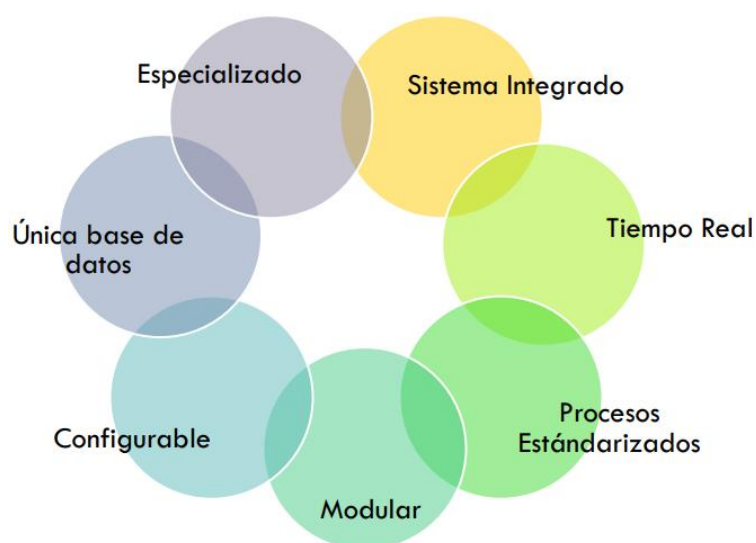
Antes, para cada una de las distintas áreas que puede ofrecer una empresa, existía un Software diferente con diferentes bases de datos, por lo tanto, una misma información se encontraba repetida por distintas bases de datos, ahora, gracias a los ERP, nos permite tener un único Software con una única base de datos y con diferentes aplicabilidades y toda la información se encuentra en un único sitio.

Una vez entendido el concepto podemos decirlo de la siguiente forma:

- Un ERP permite contar con un sólo programa de software que satisfaga las necesidades de todos los departamentos de la empresa. Antiguamente cada uno de los sectores contaba con su propio sistema. Lo que el ERP hace es, combinar todos los sistemas en un solo programa de software integrado que “corre” (ejecuta) en una sola base de datos, de tal manera que varios departamentos puedan intercambiar, acceder y actualizar información y comunicarse con los otros departamentos más fácilmente.

Los sistemas ERP típicamente manejan la producción, logística, distribución, inventario, envíos, facturas y contabilidad de la compañía de forma modular. Sin embargo, en el software ERP puede intervenir en el control de muchas actividades de negocios como ventas, entregas, pagos, producción, administración de inventarios, calidad y recursos humanos. Lo bueno de todo esto es que en cualquier momento se le puede añadir un área más al sistema ERP sin problemas, por ejemplo, una empresa que acaba de iniciar su actividad puede contar con un ERP de 4 áreas diferentes, pero dentro de un tiempo puede añadir todas las áreas que quiera.

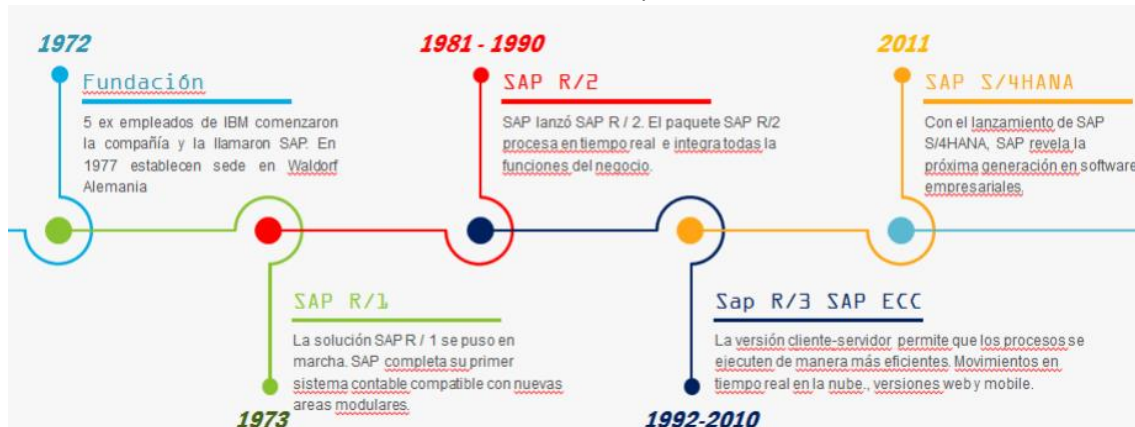
Sus características son:



2. ¿Qué es SAP?:

SAP (Systems, Applications and Products in data processing → Sistemas, Aplicaciones y Productos en el Procesamiento de datos) es un ERP muy conocido que proporciona soluciones a pequeñas, medianas y grandes empresas. Es un sistema en línea que coordina la estructura y los procesos de todos los departamentos en tiempo real. Existen diferentes versiones dependiendo de la empresa que quiera implantar este sistema.

Línea del tiempo:

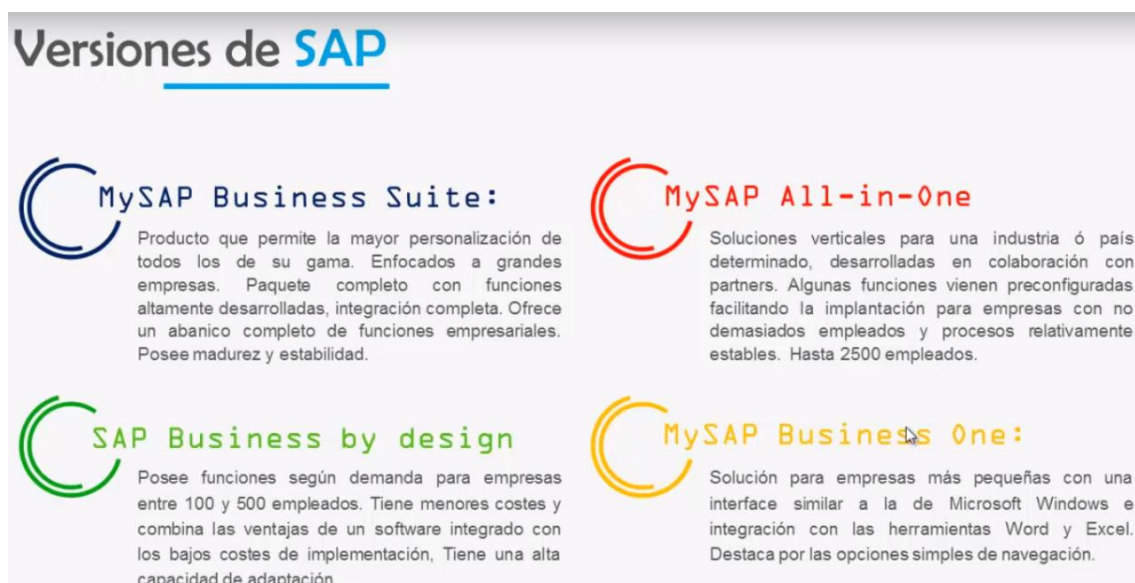


Algunas de sus características son:

- Multicompañía, multi-idioma, multimoneda.
- Plataforma escalable e integrada en tiempo real.
- Software localizado a la normativa legal de los países: retenciones de impuestos, libros de compras / ventas, ajustes por inflación, nómina.
- Soporte de múltiples estructuras organizativas y procesos empresariales estandarizados, que ofrecen soluciones específicas a medida de la empresa.
- Uso a nivel mundial.

2.1. Versiones de SAP:

Podemos distinguir varios productos de SAP dependiendo de la empresa a la que está enfocada la solución:



Tipos de industrias donde SAP se aplica:

Aerospace/Defence	Education
Automotive	Insurance
Banking	Oil & Gas
Chemicals	Pharmaceuticals
Consumer Products	Public Sector
Construction	Retail
Financial Provider	Services
Healthcare	Telecommunication

3. Arquitectura de un Sistema SAP:

La arquitectura de SAP es abierta, funciona en todo tipo de ordenadores y en los distintos tipos de sistemas operativos. Su arquitectura es escalable, gracias a su arquitectura **Cliente/Servidor** con **3 niveles distintos**:

1. **Servidor de base de Datos:** es el ordenador central que gestiona todas las funciones de la base de datos, entre ellas la consulta, la modificación, la inserción o eliminación de datos (Donde están los datos).
2. **Servidor de aplicaciones:** Está conectado al servidor de Base de Datos, y para cada departamento de la empresa, carga y ejecuta los programas y aplicaciones. Se instala en el cliente o en la nube. Es donde se desarrolla y se prueba los programas (Obtienen los datos de la base de datos).
3. **Servidor de presentación:** Simplemente es la interfaz, en los diferentes ordenadores de la empresa o personales, que se conectan al servidor de aplicaciones que presentan y hacen accesible la información y los procesos al usuario. (SAPGUI: interfaz de usuario final, SAP Graphical user Interface) (Presentan los datos al usuario).



SAP se compone de ambientes y cada ambiente por Mandantes, el conjunto de la cadena de ambientes se le conoce como **Landscape** de SAP, normalmente en la implementación se mantiene la misma estructura en la que **se trabaja con 3 servidores**:

1. **Servidor de Desarrollo**: se usa para desarrollar los programas, las correcciones, las configuraciones..., se desarrolla el programa y se crean las configuraciones necesarias para poder utilizarlo. Las configuraciones se transportan al siguiente servidor con la llamada "Orden de transporte".
2. **Servidor de Calidad**: se usa para realizar las diferentes pruebas de los diferentes programas creados en el servidor de desarrollo, se prueba tanto el programa como las configuraciones necesarias.
3. **Servidor de Producción**: una vez creado el programa y probado se utiliza este servidor, que es donde la aplicación se "lanza" de forma oficial y es donde se hace las operaciones necesarias.

Resumen: primero se crea el programa, se prueba y una vez que funciona todo correctamente se produce de forma oficial. Las configuraciones se transportan al siguiente servidor con una orden de transporte.



Esquema:



Se usa en el desarrollo de programas, correcciones, adaptaciones, configuraciones, pruebas unitarias, entre otros.



Se usa para realizar las diferentes pruebas de los diferentes programas, correcciones y adaptaciones que se realizan en desarrollo, previo a esto se realiza el transporte de las mismas.



Se realiza el transporte a este ambiente, cuando las pruebas han sido concluidas y aceptadas por el control de calidad.

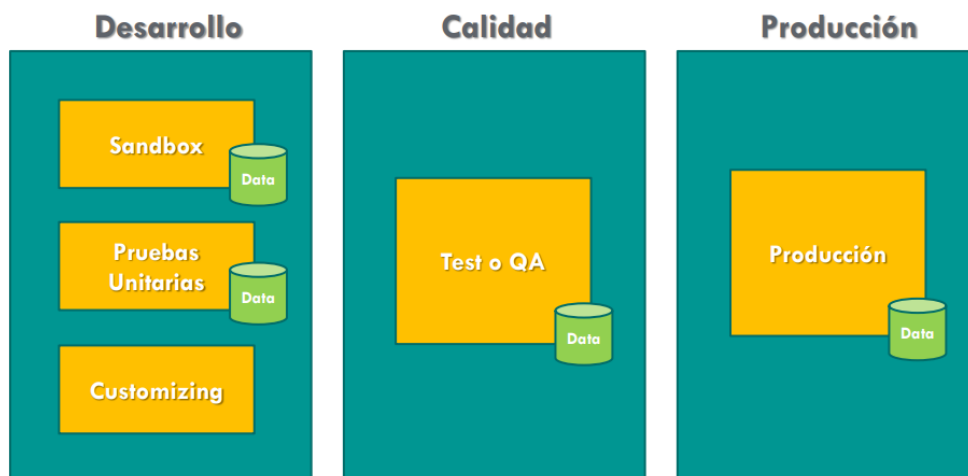
Dentro de un ambiente, por ejemplo, el ambiente de desarrollo puede existir varios ambientes, que se conocen con el nombre de Mandante, por lo tanto, en un mismo ambiente pueden existir varios mandantes. Dicho mandante es el ambiente de trabajo en el cual el usuario desarrolla su tarea.

Por ejemplo: estamos en el ambiente de desarrollo, y como usuario utilizo:

- **Mandante 100:** para desarrollar el programa y su configuración.
- **Mandante 200:** pruebo el programa en el mismo ambiente de desarrollo, realizo las pruebas unitarias.
- **Mandante 300:** pruebo las configuraciones y el desarrollo del programa, por si falla la prueba del programa con la prueba de la configuración.

Todos estos mandantes tienen en común una zona, llamada **Zona Workbench**, es donde se quedan almacenados los programas desarrollados.

Ampliando el concepto de **Landscape**, es el conjunto de la cadena de ambientes con el conjunto de mandantes que existen en cada ambiente, es decir, es el esquema de sistemas y mandantes. Ejemplo de Landscape:



Podemos observar los 3 servidores, donde en cada servidor hay un ambiente (el color azul), y este a su vez está dividido en uno o varios mandantes (el color amarillo).

3.1. SAP GUI:

SAP es una aplicación distribuida, donde se utiliza el software de cliente (SAPGUI) instalado en los equipos de trabajo de un usuario para acceder al servidor central de SAP remotamente a través de la interfaz SAP GUI.



SAP GUI es el programa estándar para acceder a casi todas las soluciones SAP. Es la plataforma utilizada para el acceso remoto al servidor central SAP en una red de empresa y permite al usuario acceder a la funcionalidad de SAP y sus aplicaciones.

4. Principales módulos SAP:

SAP está formado por varios módulos de aplicación que soportan todas las transacciones de negocios de la empresa y están integrados en forma interactiva, por lo tanto, cualquier cambio en los datos de un módulo automáticamente modificará los datos en los módulos que estén involucrados dichos datos.

Todos los módulos de aplicación tienen una arquitectura común y la misma interface con el usuario.

Los principales módulos son:

FI Finanzas	CO Costos	SD Ventas y Distribución
MM Manejo de Materiales	PM Mantenimiento de Planta	PP Planificación de la Producción
QM Gestión de la Calidad	HR Recursos Humanos	ABAP

Con todos estos módulos, SAP permite poder gestionar y controlar las diferentes áreas de una empresa.

El módulo ABAP es especial ya que:

-Es el Módulo que permite realizar las adaptaciones del sistema a los requerimientos muy específicos del cliente (formularios, cargas de datos, reportes, transacciones, etc.).

-La programación es en lenguaje ABAP (Advance Business Application Programming), perteneciente a SAP, utilizando SQL como herramienta de acceso a Base de Datos y basado en lenguajes como Pascal y C++. Adicionalmente ofrece la versatilidad de programar orientado a objetos.

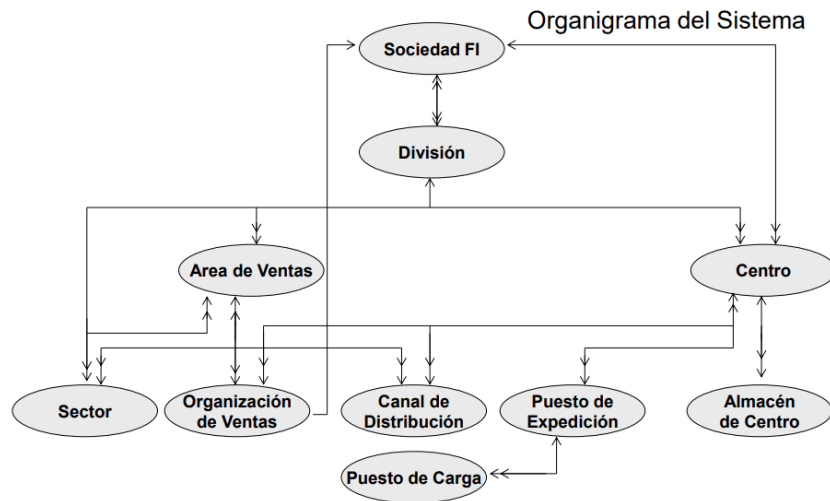
5. Conceptos Básicos:

- **Customizing:** Es la configuración del sistema necesaria para representar la estructura legal y los procesos de negocio de la empresa.
- **Unidades Organizativas:** Representa la estructura jerárquica de la empresa en el sistema. Se plasma la estructura real de la empresa. Los distintos departamentos, las divisiones...
- **Datos Maestros:** Corresponden a los datos necesarios para poder realizar transacciones del proceso en el sistema. Por ejemplo, clientes, materiales, proveedores, precios, etc.
- **Batch Input:** Es un método de SAP que permite la entrada de grandes o masivas cantidades de registros desde un sistema antiguo, desde un proceso de SAP o desde una interface. Por ejemplo, tenemos todos los datos en una hoja de Excel y lo queremos pasar a SAP.
- **Documentos:** Cada transacción de negocio que registra datos en la base de datos crea un documento con un número de identificación único. No se pueden borrar del sistema, ya que si procedemos a borrarlos podemos estropear los datos.
- **Workflow:** Es una herramienta de Soporte que puede ser utilizada para optimizar la ejecución de las actividades realizadas en el sistema. Permite automatizar la coordinación del flujo de procedimientos y de ejecución de los procesos.
- **"Z":** En el mundo de SAP, en muchas oportunidades, se requiere de desarrollos propios, reportes e informes propios, que SAP no ha contemplado, por ser específicos del usuario del sistema. Estos Programas, se les llama "Z".
- **Transacción:** Son las claves que ejecutan procesos de negocio en el sistema SAP. Por ejemplo, crear pedidos de venta, modificar dato maestro de un cliente o visualizar un reporte.
- **Sistemas de Información:** Transacciones, documentos, datos maestros son capturados y almacenados en SAP. Diversos tipos de análisis pueden ser ejecutados en cualquier momento y en tiempo real.
- **Modo SAP:** Se define como "MODO" a la ventana principal de trabajo en SAP. Son instancias de SAP.
- **ODATA:** servicio para poder realizar peticiones a una API que existe en el sistema SAP, en esta API están toda la información guardada en las tablas y sus derivados.

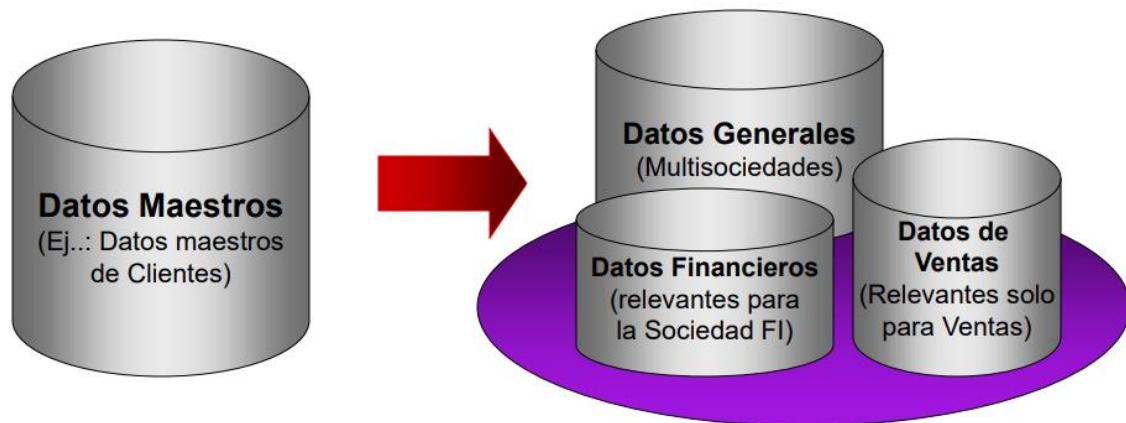
5.1. Estructura Organizativa:

-Estructura Organizativa: es el conjunto de entidades que reflejan el modelo organizativo de la empresa en SAP. La unidad básica de la estructura organizativa en SAP es el mandante o CLIENT, ya que diferencia una instancia de SAP de otra (aun cuando compartan el mismo servidor).

Organigrama de un sistema SAP:



-Los Datos Maestros: son aquellos datos que tienden a ser poco cambiantes y que definen las distintas características de los principales actores de un sistema ERP y sus correspondientes módulos. Por ejemplo, los clientes (acreedores), los proveedores, los productos o materiales (terminados, insumos, en proceso, suministros, etc.), los equipos... ya que sin estos datos no podemos realizar ninguna operación en el sistema SAP.



En Resumen:

SAP: es un sistema modular en la que cada módulo se encarga de aplicar una solución determinada a un área empresarial específica.

Podemos destacar los siguientes módulos dentro de SAP:

- ✓ **Gestión financiera (FI).** Libro mayor, libros auxiliares, ledgers especiales, etc.
- ✓ **Controlling (CO).** Gastos generales, costes de producto, cuenta de resultados, centros de beneficio, etc.
- ✓ **Tesorería (TR).** Control de fondos, gestión presupuestaria, etc.
- ✓ **Sistema de proyectos (PS).** Grafos, contabilidad de costes de proyecto, etc.
- ✓ **Gestión de personal (HR).** Gestión de personal, cálculo de la nómina, contratación de personal, etc.
- ✓ **Mantenimiento (PM).** Planificación de tareas, planificación de mantenimiento, etc.
- ✓ **Gestión de calidad (QM).** Planificación de calidad, inspección de calidad, certificado de calidad, aviso de calidad, etc.
- ✓ **Planificación de producto (PP).** Fabricación sobre pedido, fabricación en serie, etc.
- ✓ **Gestión de material (MM).** Gestión de stocks, compras, verificación de facturas, etc.
- ✓ **Comercial (SD).** Ventas, expedición, facturación, etc.
- ✓ **Workflow (WF), Soluciones sectoriales (IS),** con funciones que se pueden aplicar en todos los módulos.

6. ABAP:

Como ya hemos dicho anteriormente, ABAP es el lenguaje de programación de SAP, es un lenguaje de cuarta generación con las siguientes características:

- Utiliza sentencias de **OPEN SQL** para el acceso a la base de datos.
- Proporciona ayuda sobre la semántica y sintaxis utilizada en su lenguaje.
- Permite conexiones RFC (Remote Functions Call) para conectarse con sistema SAP o no SAP.
- Permite mostrar, con gran facilidad, reportes para presentar información por pantalla.
- Dispone de la opción de crear formularios rápidamente mediante diferentes alternativas: Sapscripts, Smartforms y Adobeforms.
- Mediante la plataforma NetWeaver, se pueden crear aplicaciones web (BSP, Web Dynpro, SAP UI5 y fiori)
- Dispone de diferentes opciones para extender las funciones estándar de SAP, entre ellas: Field exits, User exits, Badis, Enhancement, etc.
- Al ser un lenguaje de programación tipado, nos encontramos con los siguientes tipos:

Predefined ABAP Data Types

Type	Description	Initial Value	Length
C	Character	Space	1 – 65535
D	Date	'00000000'	8 characters
F	Floating Point	0.0	8 bytes
I	Integer	0	4 bytes
N	Numeric Text	'0'	1 – 65535
P	Packed Decimal	0	1 – 16 bytes
T	Time	'000000'	6 characters
X	Hexadecimal	'00'	1 – 65535
String	Variable-length	Space	Variable
xstring	Variable-length Hexadecimal	Blank string	Variable

6.1. Transacciones:

Como ya hemos dicho anteriormente, las transacciones son las claves con las que se identifican un programa, es decir, yo desarrollo un programa, y para acceder a ese programa debería de navegar por todo el sistema SAP hasta encontrar dicho programa desarrollado, entonces para aliviar ese proceso, se le asigna una clave a un programa, y escribiendo esa clave (Comando) en el campo de las transacciones accedes directamente al programa, sin buscarlo en el sistema SAP, es como un acceso directo. Las más utilizadas son las siguientes:

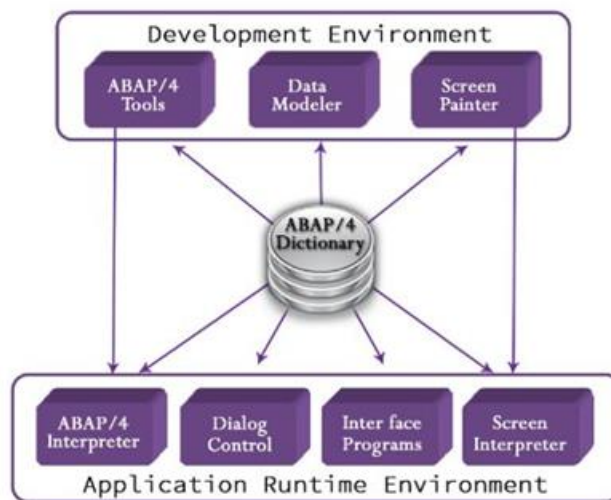
CÓDIGOS DE TRANSACCIÓN:	DESCRIPCIÓN:
SE51	Para ver o editar la Dynpro (Interface).
SM30	Para actualizar las vistas de las tablas base de datos.
SE24	Generador de Clases ABAP. (Importante)
SHDB	Registros de transacciones.
SE78	Gestor de gráficos de los formularios.
SMARTFORMS	Formularios modernos y bonitos.
SE16N	Para visualizar datos de una base de datos. (Importante)
SE91	Registro de mensajes de errores, información,...
SE37	Módulos de funciones ABAP. (Importante)
SE93	Actualizar o asignar código de transacción.
SA38	Para reportar un programa ABAP.
SE80	Navegador de objetos. (Importante)
SE38	Para desarrollar los programas. (Importante)
SE11	Es el diccionario de ABAP, para crear base de datos, elementos de datos, estructuras... (Importante)
SE14	Utilidades para base de datos, para tratar con ellas. (Importante)
AL11	Servidor de SAP con sus ficheros.
CG3Z	Subir fichero de Local al Servidor SAP.
CG3Y	Descargar fichero de Servidor a Local.
XK01, XK02, XK03	Dar de alta, modificar o visualizar a Proveedores.
XD01, XD02, XD03	Dar de alta, modificar o visualizar a Clientes.
KNA1	Visualizar Todos los Clientes.
MM01,MM02,MM03	Crear, modificar y visualizar Materiales.
SM35	Juego de datos (Sesiones de Batch Input).
SM37	Selección de JOB simples (Trabajos automatizados).
SE41	Menu Painter, para ver, modificar o copiar status de Dynpro.
ST22	Visualizar DAM. (Todos los fallos ocurrido en el servidor) .
SNUM	Creador de rangos.
SEGW	SAP Gateway Service Builder, control Odata

6.2. Diccionario de datos:

Es una de las características más importantes de ABAP, ya que, el diccionario de datos es la fuente central de información del sistema, es decir, contiene todos los datos almacenados. Todo lo que se realice en el sistema, ya sea programas, tablas internas, estructuras, elementos de datos, dominios, etc. se guarda aquí.

Su objetivo es la creación y administración de los datos (metadatos).

INTEGRATION INTO ABAP/4 WORKBENCH



Dentro del diccionario podemos encontrar las siguientes herramientas:

Dictionary ABAP: Imagen inicial

se11

Dictionary ABAP: Imagen inicial

☒ Tabla base datos

☐ Vista

☐ Tipo de datos

☐ Grupo tipos

☐ Dominio

☐ Ayuda p.búsqueda

☐ Objeto de bloqueo

Visualizar Modificar Crear

- ❖ **Dominios:** Define el tipo de datos de un campo (CHAR, NUM, I, DEC...) y su longitud. En algunos casos, también delimitan los valores posibles del campo mediante ámbito de valores fijos (Dándole los valores posibles como en las enumeraciones (JAVA)). Se crean en la transacción SE11.

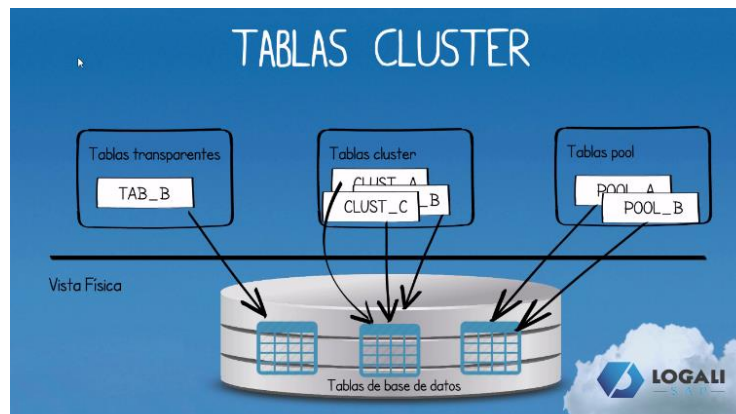
Dominio	ZMPBEXAMEN	activo
Descripción breve	dominio para el resultado del examen, char100	
Atributos	Definición	Ámbito val.
Formato		
Tipo de datos	CHAR	String
Ctd.posiciones	100	
Decimales	0	
Propiedades salida		
Longitud salida	100	
Rutina conv.		
<input type="checkbox"/> Signo +/-		
<input checked="" type="checkbox"/> Minúsculas		

- ❖ **Elementos de datos:** para su creación, necesitan un dominio, define la representación semántica de un campo. Una variable está definida por una representación semántica, por ejemplo, la variable nombre, para utilizarla, necesita de un elemento de datos que le dé el nombre de la representación, que esta a su vez, necesita un dominio que es el que dice que la variable nombre es de tipo char. Se crean en la transacción SE11.

Elemento datos	Z_MPB_CHAR_EXAMEN	activo
Descripción breve	Resultado	
Atributos	Tipo datos	Propiedades adicionales
<p>Tipo elemental</p> <p><input checked="" type="radio"/> Dominio</p> <p>ZMPBEXAMEN dominio para el resultado del e...</p> <p>Tipo datos CHAR String</p> <p>Longitud 100</p> <p><input type="radio"/> Tipo instalado</p> <p>Tipo datos</p> <p>Longitud 0</p> <p><input type="radio"/> Tipo referencia</p> <p><input type="radio"/> Tipo referenciado</p> <p><input type="radio"/> Referencia a tipo instalado</p> <p>Tp.datos</p> <p>Longitud 0</p>		
Dict: Visualizar elemento datos		
<p>Dict: Visualizar elemento datos</p> <p>Documentación</p>		
Elemento datos	Z_MPB_CHAR_EXAMEN	activo
Descripción breve	Resultado	
Atributos	Tipo datos	Propiedades adicionales
Denom.campo		
Breve	Long.	Denominador de campo
Mediano	9	Resultado
Largo	16	Resultado Examen
Cabecera	21	Resultado para examen

❖ **Tablas:** son las fuentes de información, es decir, son objetos donde se almacenan los datos de forma persistente (Tablas de BASE DE DATOS). Están compuestas por campos claves que identifican de manera unívoca cada entidad que se almacena en ella. Se crean en la transacción SE11. Hay diferentes tipos de tablas:

- **Tablas transparentes:** son las tablas físicas dentro del sistema de base de datos, utilizadas para almacenar los datos empresariales y de aplicación que se usan dentro de SAP.
- **Tablas cluster:** se almacenan en un cluster de Base de datos. Tabla de base de datos que contiene a su vez varias tablas.
- **Tablas pool:** almacenan la información en una tabla física dentro del gestor de base de datos. Varias tablas del diccionario de datos se corresponden con una sola tabla de base de datos (tabla pool). Se utilizan para almacenar miles de pequeñas tablas de la capa de aplicación en pocas tablas de base de datos



Ejemplos de tablas:

- Campos de la tabla SCARR (BASE DE DATOS Compañías Aéreas):

Dict: Visualizar tabla									
Dict: Visualizar tabla									
Tabla transparente SCARR activo									
Descripción breve Compañía aérea									
Atributos Entrega y actualización Campos Ayuda p./Verif.entr. Campos de moneda/cantidad									
Ay.búsq. Tipo instalado 1 / 5									
Campo	Civ	Val...	Elem.datos	Tipo de d...	Long.	Deci...	Descripción breve		Grupo
MANDT	✓	✓	S_MANDT	CLNT	3		0 Mandante		
CARRID	✓	✓	S_CARR_ID	CHAR	3		0 Denominación breve de la compañía aérea		
CARRNAME	✓	✓	S_CARRNAME	CHAR	20		0 Nombre de una compañía aérea		
CURRCODE	✓	✓	S_CURRCODE	CURY	5		0 Moneda local de la compañía aérea		
URL	✓	✓	S_CARRURL	CHAR	255		0 URL de una compañía aérea		

- DATOS:

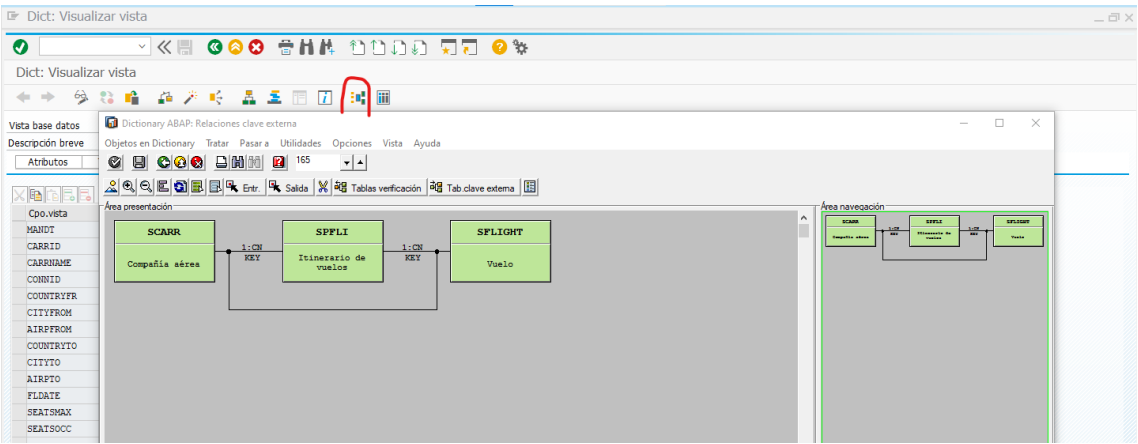
Data Browser: Tabla SCARR 18 aciertos

Data Browser: Tabla SCARR 18 aciertos

Tabla verificación

MAN...	CARR...	CARRNAME	CURCODE	URL
100	AA	American Airlines	USD	http://www.america.com
100	AB	Air Berlin	EUR	http://www.airberlin.de
100	AC	Air Canada	CAD	http://www.aircanada.ca
100	AF	Air France	EUR	http://www.airfrance.fr
100	AZ	Alitalia	EUR	http://www.italia.it
100	BA	British Airways	GBP	http://www.british-airways.com
100	CO	Continental Airlines	USD	http://www.continental.com
100	DL	Delta Airlines	USD	http://www.delta-air.com
100	FJ	Air Pacific	USD	http://www.airpacific.com
100	JL	Japan Airlines	JPY	http://www.jal.co.jp
100	LH	Lufthansa	EUR	http://www.lufthansa.com
100	NG	Lauda Air	EUR	http://www.laudaair.com
100	NW	Northwest Airlines	USD	http://www.nwa.com
100	QF	Qantas Airways	AUD	http://www.qantas.com.au
100	SA	South African Air.	ZAR	http://www.saa.co.za
100	SQ	Singapore Airlines	SGD	http://www.singaporeair.com
100	SR	Swiss	CHF	http://www.swiss.com
100	UA	United Airlines	USD	http://www.ual.com

- Mapa de relación respecto a otras tablas de BASE DE DATOS:



- ❖ **Estructuras:** es el conjunto de campos almacenados en una tabla. Ejemplo tenemos una tabla cliente con los campos: Nombre, Apellidos, DNI, Teléfono. Pues una estructura es un tipo de tabla con los campos Nombre, Apellidos, DNI, Teléfono. Es decir, lo único que se va a poder guardar en esta estructura debe tener esa información. Es una tupla de la tabla de BASE DE DATOS. Se crean en la transacción SE11.

Dict: Visualizar estructura

Dict: Visualizar estructura

Estructura: ZMPBSTRUCTEXAMEN activo

Descripción breve: estructura ejercicio alv

Atributos Componentes Ayuda p./Verif.entr. Campos de moneda/cantidad

Componente	Clase tipificación	Tp.componente	Tipo de d...	Long.	Deci...	Descripción breve	Grupo
BUKRS	1 Type	BUKRS	CHAR	4	0	Sociedad	
KTOKD	1 Type	KTOKD	CHAR	4	0	Grupo de ctas.deudor	
NAME1	1 Type	AD_NAME1	CHAR	40	0	Nombre 1	
STRAS	1 Type	AD_STREET	CHAR	60	0	Calle	
POSTL	1 Type	AD_POSTCD1	CHAR	10	0	Código postal de la población	
LAND1	1 Type	LAND1	CHAR	3	0	Clave de país	
REGIO	1 Type	REGIO	CHAR	3	0	Región (Estado federal, "land", provincia, condado)	
SFRAS	1 Type	SFRAS	LANG	1	0	Clave de idioma	
TELF1	1 Type	AD_TLNMBR1	CHAR	30	0	Primer número teléfono: Prefijo + número	
RESULTADO	1 Type	Z_MPB_CHAR_EXAM...	CHAR	100	0	Resultado	

Por Ejemplo, la tabla ZMPB_CLIENTES es la siguiente:

Dict: Visualizar tabla

Dict: Visualizar tabla

Tabla transparente: ZMPB_CLIENTES activo

Descripción breve: tabla para realizar ejercicios prácticos

Atributos Entrega y actualización Campos Ayuda p./Verif.entr. Campos de moneda/cantidad Índices

Campo	Clv	Val...	Elem.datos	Tipo de d...	Long.	Deci...	Descripción breve	Grupo
MANDT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	MANDT	CLNT	3	0	Mandante	
ID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		CHAR	5	0	Id	
NOMBRE	<input type="checkbox"/>	<input type="checkbox"/>		CHAR	20	0	Nombre	
APELLIDOS	<input type="checkbox"/>	<input type="checkbox"/>		CHAR	40	0	Apellidos	
EDAD	<input type="checkbox"/>	<input type="checkbox"/>		INT2	5	0	Edad	

Para esta tabla, he creado la siguiente estructura, que si nos fijamos no tiene todos los campos de la base de datos, esto se puede hacer perfectamente, ya que la estructura es una referencia a los campos de dicha tabla, pero no tiene que ser obligatoriamente todos los datos:

Dictionary: Modificar estructura

Dictionary: Modificar estructura

Estructura: ZMPB_ESTRUCTURA_CLIENTES activo

Descripción breve: ESTRUCTURA PARA LA TABLA ZMPB_CLIENTES

Atributos Componentes Ayuda p./Verif.entr. Campos de moneda/cantidad

Componente	Clase tipificación	Tp.componente	Tipo de d...	Long.	Deci...	Descripción breve
MANDT	1 Type	MANDT	CLNT	3	0	Mandante
ID	1 Type		CHAR	5	0	
NOMBRE	1 Type		CHAR	20	0	
APELLIDOS	1 Type		CHAR	40	0	

- ❖ **Vistas:** es una forma de visualizar y modificar, en tiempo real, el contenido de una o más tablas al mismo tiempo. Es una vista, es decir, una forma de ver los datos que contiene una tabla y de modificarlos, por lo tanto, podemos ocultar los campos que queramos que no se vean en dicha vista. Las vistas se crean en la transacción SE11 y en la SM30 se puede ver. Por Ejemplo: una vista de la tabla de datos de mensajes, esta vista tiene como objetivo informar de los datos que contiene la tabla, gestiona los mensajes.

Visualizar vista Gestión de mensajes: Resumen

Visualizar vista Gestión de mensajes: Resumen

Gestión de mensajes

Programa / Incluye	Tp.mje.	Clase de mensajes	Nº mje.	Acción	Tp.mje.	Clase de mensajes	Nº mje.
ASIGNA_OPER	A	CPCC_DM	60	0 No actuar			0
ASIGNA_OPER	E	CPCC_DM	2	0 No actuar			0
ASIGNA_OPER	E	V2	2	1 Reemplazar	A	ZM401	1
ASIGNA_OPER	I	CPCC_DM	16	0 No actuar			0
ASIGNA_OPER	I	CPCC_DM	20	0 No actuar			0
ASIGNA_OPER	I	CPCC_DM	50	0 No actuar			0
ASIGNA_OPER	S	CPCC_DM	18	0 No actuar			0

- ❖ **Ayudas de búsquedas:** es un objeto para definir posibles valores de ayuda de un campo de una tabla de base de datos o de un campo de pantalla. Se utiliza mucho para darle una ayuda al usuario, sin esta ayuda, el usuario tendría que poner, a mano, un dato en específico y podría causar algún tipo de error, ya que se podría confundir. Por Ejemplo, en la siguiente imagen, se puede observar un cuadradito al lado del Input de Sociedad, eso hace referencia a la ayuda de búsqueda.

ejercicio de alv de la tabla EKKO

Introduce los siguientes datos:

Documento compras	<input type="text"/>	a	<input type="text"/>	
Sociedad	<input type="text"/>	a	<input type="text"/>	
Teléfono	<input type="text"/>	a	<input type="text"/>	
Centro suministrador	<input type="text"/>	a	<input type="text"/>	
Creado el	<input type="text"/>	a	<input type="text"/>	
Proveedor	<input type="text"/>	a	<input type="text"/>	
Licitación	<input type="text"/>	a	<input type="text"/>	
Nº dirección	<input type="text"/>	a	<input type="text"/>	

Si lo pulsamos, nos saldrá una pantallita con los valores que podemos insertar en ese campo, evitando que el usuario se confunda:

ejercicio de alv de la tabla EKKO

Introduce los siguientes datos:

Documento compras	<input type="text"/>
Sociedad	<input type="text"/>
Teléfono	<input type="text"/>
Centro suministrador	<input type="text"/>
Creado el	<input type="text"/>
Proveedor	<input type="text"/>
Licitación	<input type="text"/>
Nº dirección	<input type="text"/>

Sociedad (1) 9 Entradas encontradas

Sociedades

Soc.	Nombre de la empresa	Población	Mon.
004	Rodin, S.A.U.	Torello	EUR
005	S&P Sistemas de Vent.	Parets del Valles	EUR
006	Electromecánicas MC S.A.U	Sils	EUR
007	Ventiladores Chaysol, SAU	Pinto	EUR
009	S&P FRANCE S.V.	Thuir	EUR
050	S&P Ventilation Group SLU	Barcelona	EUR
051	S&P Research, S.L.U.	Parets del Valles	EUR
101	S&P Industries, S.L.U.	Ripoll	EUR
Z008	EMPRESA PRUEBA MIGRACIÓN	Thuir	EUR

- ❖ **Objetos de bloqueo:** es un método para coordinar el acceso de los usuarios a un mismo recurso, es decir, puede ser que dos usuarios estén modificando, al mismo tiempo, un programa o tabla, por lo que puede dar un fallo de recurrencia. Por eso, antes de acceder a los datos, se debe de realizar el bloqueo para que ningún otro usuario pueda acceder al mismo tiempo. De la misma manera, cuando ya no se necesite el acceso a esos datos, se deberá de desbloquear dicho objeto.

The screenshot displays the SAP Dictionary 'Display Lock Object' interface. At the top, there are radio buttons for 'Database table', 'View', 'Data type', 'Type Group', 'Domain', 'Search help', and 'Lock object'. The 'Lock object' option is selected, and the value 'EZ_ZHR_ELS_021' is entered in the adjacent field. Below these fields are three buttons: 'Display', 'Change', and 'Create'. The main section is titled 'Dictionary: Display Lock Object' and contains a toolbar with various icons. Below the toolbar, the 'Lock object' field shows 'EZ_ZHR_ELS_021' and its status is 'Active'. The 'Short Description' field contains the text 'Objeto de bloqueo para la tabla ZHR_ELS_021'. There are three tabs: 'Attributes', 'Tables', and 'Lock parameter'. The 'Lock parameter' tab is currently selected, showing a 'Primary Table' section with 'Name' set to 'ZHR_ELS_021' and 'Lock Mode' set to 'Write Lock'.

7. ¿Cuál es la estructura de un programa?:

Antes de empezar, debemos tener claro que un programa, ya sea ejecutable o no ejecutable, en SAP, se llama **Report**.

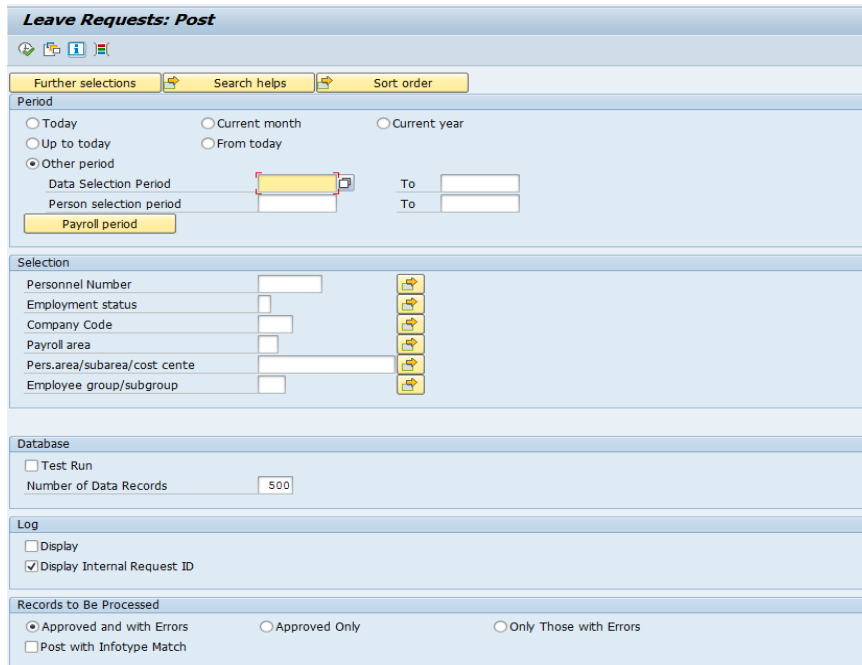
Existen diferentes programas o Report en SAP:

- **Report ejecutable:** es un programa ejecutable, no necesita un programa externo para ser ejecutado. Es el programa principal que contiene report include.
- **Report include:** es un programa no ejecutable. Contiene código que podrá ser ejecutado desde otro report externo (report principal ejecutable).
- **Modulpool:** programa con control de pantallas, con más complejidad que un programa ejecutable, pero ya es un poco antiguo.

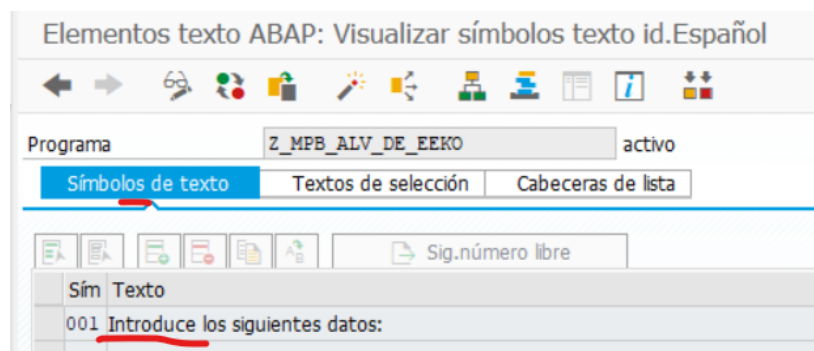
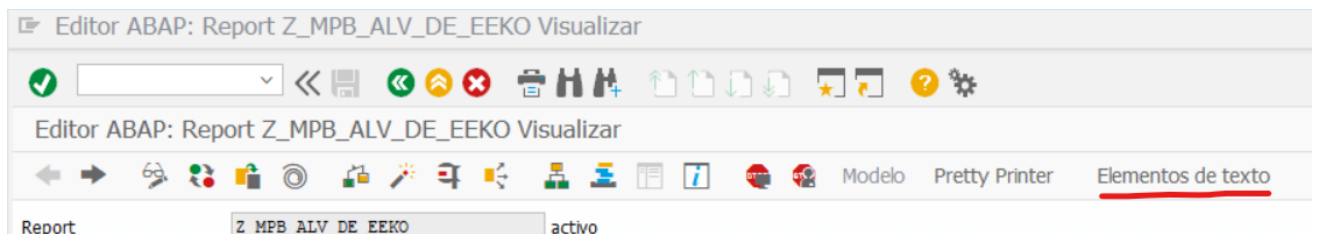
Diseñado por: **MARCOS PUERTO BARRERO**

Un Report (Programa) contiene las diferentes partes:

- ✓ **La pantalla de selección:** es la pantalla principal donde el usuario debe ingresar, en los campos, los datos que se le piden.



- ✓ **Elementos de textos:** son textos necesarios para nombrar elementos de la pantalla de selección, mensajes, columnas en los listados, etc. Suelen llevar asociados traducciones, de modo que, en función del lenguaje con el que el usuario acceda a SAP se visualizará en un idioma o en otro.



Elementos texto ABAP: Visualizar textos selección id.Español

Programa **Z_MPB_ALV_DE_EEKO** activo

Símbolos de texto **Textos de selección** Cabeceras de lista

Nombre	Texto	Referencia...
SO_ADRNR	Nº dirección	<input checked="" type="checkbox"/>
SO_AEDAT	Creado el	<input checked="" type="checkbox"/>
SO_BUKRS	Sociedad	<input checked="" type="checkbox"/>
SO_EBELN	Documento compras	<input checked="" type="checkbox"/>
SO_LIFNR	Proveedor	<input checked="" type="checkbox"/>
SO_RESWK	Centro suministrador	<input checked="" type="checkbox"/>
SO_SUBMI	Licitación	<input checked="" type="checkbox"/>
SO_TELF1	Teléfono	<input checked="" type="checkbox"/>

En el programa se hace referencia de la siguiente forma:

```
8
9  REPORT y_test_report.
10
11  TABLES: pa0001.
12
13  SELECTION-SCREEN BEGIN OF BLOCK b01 WITH FRAME TITLE text-001.
14  |
15  PARAMETERS: p_pernr TYPE persno .
16  SELECT-OPTIONS: so_bukrs FOR pa0001-bukrs DEFAULT '12345' .
17
18  SELECTION-SCREEN END OF BLOCK b01.
```

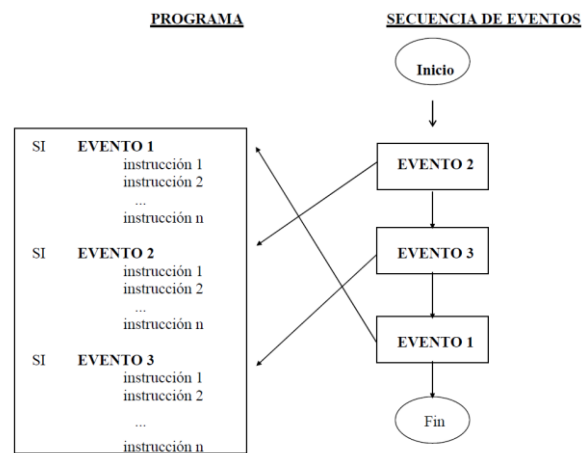
ejercicio de alv de la tabla EKKO

ejercicio de alv de la tabla EKKO

Introduce los siguientes datos:

Documento compras	<input type="text"/>	a	<input type="text"/>	<input type="button" value="icon"/>
Sociedad	<input type="text"/>	a	<input type="text"/>	<input type="button" value="icon"/>
Teléfono	<input type="text"/>	a	<input type="text"/>	<input type="button" value="icon"/>
Centro suministrador	<input type="text"/>	a	<input type="text"/>	<input type="button" value="icon"/>
Creado el	<input type="text"/>	a	<input type="text"/>	<input type="button" value="icon"/>
Proveedor	<input type="text"/>	a	<input type="text"/>	<input type="button" value="icon"/>
Licitación	<input type="text"/>	a	<input type="text"/>	<input type="button" value="icon"/>
Nº dirección	<input type="text"/>	a	<input type="text"/>	<input type="button" value="icon"/>

- ✓ **Eventos:** SAP, concretamente el lenguaje ABAP, no es un lenguaje con estructura lineal, es orientado a eventos. Esto quiere decir que la secuencia de instrucciones depende del evento que se haya lanzado en cada momento.



Los eventos más utilizados son los siguientes:

EVENTO	ACCIÓN
INITIALIZATION	Se lanza en el momento que se ejecuta el programa, se ejecuta antes que el “pintado” de la pantalla de selección. Sólo se ejecuta una vez y se utiliza, normalmente, para inicializar o limpiar las variables necesarias en la pantalla de selección.
START-OF-SELECTION	Se lanza cuando se ejecuta el programa desde la pantalla de selección. Es el lugar indicado para que se realice la selección de datos en función de los parámetros indicados en la pantalla de selección.
END-OF-SELECTION	Se ejecuta justo después del START-OF-SELECTION. Es el punto indicado para realizar la impresión del listado por pantalla de los datos seleccionados.
TOP-OF-PAGE	Se ejecuta en el momento previo a imprimir la página actual. Se utiliza para escribir cabeceras o títulos dentro de una página.
AT SELECTION-SCREEN	Se lanza después de que el sistema haya procesado la pantalla de selección. Se utiliza para realizar las validaciones de los campos introducidos en la pantalla de selección, si se han introducido mal, no permitirá seguir con el programa hasta que el usuario lo corrija.
AT SELECTION-SCREEN ON [parámetro/select-options]:	Es exactamente igual que el AT SELECTION-SCREEN, pero este se lanza cuando el sistema haya procesado el parámetro indicado. Se usa para validar un parámetro en concreto.
AT SELECTION-SCREEN ON VALUE-REQUEST FOR [parámetro/select-options]:	Permite crear un bloque de proceso asociado en el momento en el que se ejecuta la ayuda de búsqueda de un campo. Se muestra un pop-up con los valores posibles a seleccionar.
AT SELECTION-SCREEN ON RADIOBUTTON GROUP [radiobutton group]	Se lanza cuando se pulsa alguno de los radiobutton asociados al radiobutton group indicado. Se utiliza para validar los Radio Buttons.
AT SELECTION-SCREEN ON BLOCK [bloque]	Permite activar un bloque de proceso cuando el sistema termina de procesar un bloque en concreto. Se utiliza para validar un bloque de código concreto.
AT SELECTION-SCREEN OUTPUT	Se lanza en el momento previo en el que el sistema muestra la pantalla de selección, es decir, cada vez que se procesa la pantalla de selección se ejecuta este evento.

- ✓ **Programa:** dentro del programa principal (el ejecutable), suele estar estructurado de la siguiente forma:
 - **Programa ejecutable:** lo único que va a contener son programas includes, en el que se puede distinguir los siguientes includes:

Code listing for: Z_MPB_ALV_DE_EEKO

Description: ejercicio de alv de la tabla EKKO

```
*&-----*
*& Report Z_MPB_ALV_DE_EEKO
*&-----*
*&
*&-----*
REPORT Z_MPB_ALV_DE_EEKO.

include Z_MPB_ALV_DE_EEKO_TOP.
include Z_MPB_ALV_DE_EEKO_SEL.
include Z_MPB_ALV_DE_EEKO_EVT.
include Z_MPB_ALV_DE_EEKO_FRM.
include Z_MPB_ALV_DE_EEKO_PBO.
include Z_MPB_ALV_DE_EEKO_PAI.
```

- Nombre del programa + _TOP: este include sirve para definir todas las variables globales que se van a necesitar en el programa:

Code listing for: Z_MPB_ALV_DE_EEKO_TOP

Description: Include Z_MPB_ALV_DE_EEKO_TOP

```
*&-----*
*& Include          Z_MPB_ALV_DE_EEKO_TOP
*&-----*
TABLES: ekko.

DATA: it_ekko TYPE STANDARD TABLE OF zmpb_ekko,
      gv_mostrar TYPE flag,
      gv_mensaje TYPE string.

DATA: ok_code TYPE sy-ucomm.
*ALV
DATA: go_alvgrid TYPE REF TO cl_gui_alv_grid, "global object
      it_catalog TYPE lvc_t_fcat,
      wa_layout  TYPE lvc_s_layo.
```

- Nombre del programa + _SEL: este include sirve para definir todas las pantallas de selección:

Code listing for: Z_MPB_ALV_DE_EEKO_SEL

Description: Include Z_MPB_ALV_DE_EEKO_SEL

```
*&-----*
*& Include          Z_MPB_ALV_DE_EEKO_SEL
*&-----*
SELECTION-SCREEN BEGIN OF BLOCK bloque1 WITH FRAME TITLE TEXT-001.
SELECT-OPTIONS: so_ebeln FOR ekko-ebeln,
                 so_bukrs FOR ekko-bukrs,
                 so_telf1 FOR ekko-telf1,
                 so_reswk FOR ekko-reswk,
                 so_aedat FOR ekko-aedat,
                 so_lifnr FOR ekko-lifnr,
                 so_submi FOR ekko-submi,
                 so_adrnr FOR ekko-adrnr.
SELECTION-SCREEN END OF BLOCK bloque1.
```

- Nombre del programa + _EVT: este include sirve para definir el flujo del programa:

Code listing for: Z_MPB_ALV_DE_EEKO_EVT

Description: Include Z_MPB_ALV_DE_EEKO_EVT

```
*&-----*
*& Include          Z_MPB_ALV_DE_EEKO_EVT
*&-----*
INITIALIZATION.
  PERFORM f_limpiar_variables.

START-OF-SELECTION.
  PERFORM f_obtener_datos.

END-OF-SELECTION.
  PERFORM f_mostrar_alv.
```

- Nombre del programa + _FRM: este include sirve para encapsular toda la codificación del programa, aquí es donde realmente está el código programado:

Code listing for: Z_MPB_ALV_DE_EEKO_FRM

Description: Include Z_MPB_ALV_DE_EEKO_FRM

```
*&-----*
*& Include          Z_MPB_ALV_DE_EEKO_FRM
*&-----*
*& Form F_LIMPIAR_VARIABLES
*&-----*
*& text
*&-----*
*& --> p1          text
*& <-- p2          text
*&-----*
FORM f_limpiar_variables .
  FREE: it_ekko,it_catalog.
  CLEAR: wa_layout, gv_mensaje, gv_mostrar.
ENDFORM.
*&-----*
*& Form F_OBTENER_DATOS
*&-----*
*& text
*&-----*
*& --> p1          text
*& <-- p2          text
*&-----*
FORM f_obtener_datos .
  SELECT ebeln bukrs telf1 reswk aedat lifnr submi adrnr grwcu ernam ekorg angnr ihrez
  FROM ekko
  INTO TABLE it_ekko
  WHERE ebeln IN so_ebeln AND bukrs IN so_bukrs AND telf1 IN so_telf1
  AND reswk IN so_reswk AND aedat IN so_aedat AND lifnr IN so_lifnr AND
  submi IN so_submi AND adrnr IN so_adrnr.

  IF sy-subrc EQ 0.
    gv_mostrar = 'X'.
  ENDIF.

ENDFORM.
*&-----*
*& Form F_MOSTRAR_ALV_BASICO
*&-----*
*& text
*&-----*
```

```

FORM f_investigacion_bapi .
  DATA: purchaseorder TYPE bapimepoheader-po_number,
         poheader      TYPE bapimepoheader,
         poheaderx     TYPE bapimepoheaderx,
         return        TYPE STANDARD TABLE OF bapiret2,
         wa_return     TYPE bapiret2,
         wait          TYPE bapita-wait.

  DATA: lwa_ekko TYPE zmpb_ekko.

  LOOP AT it_ekko ASSIGNING FIELD-SYMBOL(<fs_ekko>).

    "Pasamos valores
    purchaseorder = <fs_ekko>-ebeln.

    " Recuperamos valor...
    CLEAR poheader.
    CALL FUNCTION 'BAPI_PO_GETDETAIL1'
      EXPORTING
        purchaseorder = purchaseorder
      IMPORTING
        poheader      = poheader.
    "Recuperamos el valor
    CLEAR poheader.
    CALL FUNCTION 'BAPI_PO_GETDETAIL1'
      EXPORTING
        purchaseorder = purchaseorder
      IMPORTING
        poheader      = poheader.

    "Pasamos el valor
    CLEAR return.
    CALL FUNCTION 'BAPI_PO_CHANGE'
      EXPORTING
        purchaseorder = purchaseorder
        poheader      = poheader
        poheaderx     = poheaderx
        park_complete = 'X'
      TABLES
        return        = return.

    IF sy-subrc EQ 0.
      wait = 'X'.
      CLEAR wa_return.

      CALL FUNCTION 'BAPI_TRANSACTION_COMMIT'
        EXPORTING

```

- Nombre del programa + _PBO: este include se utiliza cuando utilizamos DYNPROS diferentes, 'Process Before Output', se traduce como 'Procesos antes de la salida', sirve para realizar tareas antes de que se muestre otra pantalla diferente:

Code listing for: Z_MPB_ALV_DE_EEKO_PBO

Description: Include Z_MPB_ALV_DE_EEKO_PBO

```

*&-----*
*& Include      Z_MPB_ALV_DE_EEKO_PBO
*&-----*
*&-----*
*& Module STATUS_9000 OUTPUT
*&-----*
*&
*&-----*
MODULE status_9000 OUTPUT.
  SET PF-STATUS '9000'.
  SET TITLEBAR '9000'.
  PERFORM f_configurar_alv.
ENDMODULE.

```


- Nombre del programa + _PAI: este include se utiliza cuando utilizamos DYNPROS diferentes, 'Process After Output', se traduce como 'Procesos despues de la salida', sirve para realizar tareas después de que se muestre otra pantalla diferente:

Code listing for: Z_MPB_ALV_DE_EEKO_PAI

Description: Include Z_MPB_ALV_DE_EEKO_PAI

```
*&-----*
*& Include          Z_MPB_ALV_DE_EEKO_PAI
*&-----*
*&-----*
*&      Module  USER_EXIT_COMMAND_9000  INPUT
*&-----*
*      text
*-----*
MODULE user_exit_command_9000 INPUT.
Perform f_exit_command.
ENDMODULE.
*&-----*
*&      Module  USER_COMMAND_9000  INPUT
*&-----*
*      text
*-----*
MODULE user_command_9000 INPUT.
PERFORM f_user_command.
ENDMODULE.
```

- Nombre del programa + _LCL: este include sirve para definir todas las clases que necesitemos en un programa, esto se utiliza para poder realizar la famosa Programación Orientada a Objeto (POO):

Code listing for: Z_MPB_BECAEJERCICIO7_LCL

Description:

```
*&-----*
*& Include          Z_MPB_BECAEJERCICIO7_LCL
*&-----*
DATA: lv_index TYPE i.

CLASS lcl_eventclick DEFINITION.
  PUBLIC SECTION.

      METHODS on_dblclick FOR EVENT double_click OF cl_gui_alv_grid
        IMPORTING e_row e_column.
ENDCLASS.

CLASS lcl_eventclick IMPLEMENTATION.

  METHOD on_dblclick.
    lv_index = e_row.
    CALL FUNCTION 'POPUP_TO_INFORM'
      EXPORTING
        titel = 'Doble click'
        txt1  = 'Has pulsado en: '
        txt2  = lv_index
        txt3  = e_column
*      TXT4   = ' '
    .

  ENDMETHOD.
ENDCLASS.
```

8. Palabras Clave para programar:

- **DATA:** se utiliza para declarar variables en el programa.

```
DATA: gv_numerouno TYPE int1,  
      gv_numerodos TYPE int1,  
      gv_resultado TYPE int2.
```

- **CONSTANTS:** se utiliza para declarar constantes.

```
CONSTANTS: co_numeropi VALUE '3.1416'.
```

- **TYPES:** se utiliza para definir un tipo de estructura para poder declararnos una tabla y una estructura en base a eso.

```
□ TYPES: BEGIN OF ty_log,  
      linea      TYPE i,  
      resultado TYPE char50,  
      END OF ty_log.
```

- **TABLES:** se utiliza para declarar una tabla para poder usarla en todo el programa. Si no declaramos la tabla, las consultas que hagamos hacia ella fallarán.

```
TABLES: scarr.
```

- **Parameters:** se utiliza para introducir parámetros por teclado.

```
PARAMETERS: pa_cade TYPE string.
```

PA_CADE

hola soy marcos

- **WRITE:** se utiliza para escribir una cadena de texto o variables por pantalla.

```
WRITE: 'Hola, voy a mostrar el contenido de: ' , pa_cade.
```

```
Hola, voy a mostrar el contenido de: HOLA SOY MARCOS
```

- **SKIP N°:** se utiliza para dejar n líneas en blanco, ideal para separar datos.

```
WRITE: 'Hola, voy a mostrar el contenido de: ' , pa_cade.
```

```
SKIP 4.
```

```
WRITE: 'Acabamos de hacer una separación de 4 líneas y vamos a mostrar una línea separadora.'.
```

```
Hola, voy a mostrar el contenido de: HOLA SOY MARCOS
```

```
Acabamos de hacer una separación de 4 líneas y vamos a mostrar una línea separadora.
```

- **CLEAR:** se utiliza para inicializar/limpiar las variables.

```
CLEAR: pa_cade.
```

```
He limpiado la variable pa_cade, va a estar vacía, por lo tanto el valor es:  .
```

- **FREE:** se utiliza para limpiar una tabla.

```
FREE: scarr.
```

- **ULINE:** se utiliza para dibujar una línea por la pantalla.

```
WRITE: 'Acabamos de hacer una separación de 4 líneas y vamos a mostrar una línea separadora.'.  
CLEAR: pa_cade.  
ULINE.  
WRITE: 'He limpiado la variable pa_cade, va a estar vacía, por lo tanto el valor es: ', pa_cade , '.'
```

```
Acabamos de hacer una separación de 4 líneas y vamos a mostrar una línea separadora.
```

```
He limpiado la variable pa_cade, va a estar vacía, por lo tanto el valor es:  .
```

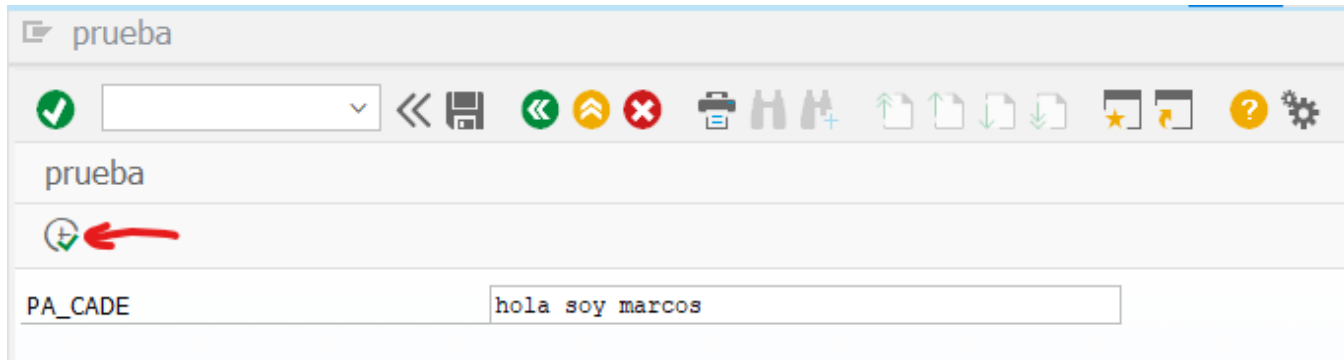
Todas las sentencias deben acabar en punto (.).

Ejemplo del programa utilizado:

```
Editor ABAP: Report Z_MPB_DOCUMENTO_PRUEBA Modificar  
Editor ABAP: Report Z_MPB_DOCUMENTO_PRUEBA Modificar  
Report Z_MPB_DOCUMENTO_PRUEBA activo  
1 *-----*  
2 *S Report Z_MPB_DOCUMENTO_PRUEBA  
3 *-----*  
4 *S  
5 *-----*  
6 REPORT z_mpb_documento_prueba.  
7  
8 DATA: gv_numerouno TYPE int1,  
9       gv_numerodos TYPE int1,  
10      gv_resultado TYPE int2.  
11  
12 CONSTANTS: co_numeropi VALUE '3.1416'.  
13  
14 TYPES: BEGIN OF ty_log,  
15       linea      TYPE i,  
16       resultado  TYPE char50,  
17     END OF ty_log.  
18  
19 TABLES: scarr.  
20  
21 PARAMETERS: pa_cade TYPE string.  
22  
23 WRITE: 'Hola, voy a mostrar el contenido de: ' , pa_cade.  
24 SKIP 4.  
25 WRITE: 'Acabamos de hacer una separación de 4 líneas y vamos a mostrar una línea separadora.'.  
26 CLEAR: pa_cade.  
27 ULINE.  
28 WRITE: 'He limpiado la variable pa_cade, va a estar vacía, por lo tanto el valor es: ', pa_cade , '.'.  
29  
30 gv_numerouno = 5.  
31 gv_numerodos = 10.  
32 gv_resultado = gv_numerouno + gv_numerodos.  
33 SKIP 1.  
34 WRITE: 'La suma de: ', gv_numerouno, ' + ', gv_numerodos, ' es: ', gv_resultado.  
35 FREE: scarr.
```

Diseñado por: **MARCOS PUERTO BARRERO**

Como vemos, al ejecutar el programa nos sale una pantalla donde tenemos que introducir los datos puestos como Parameters, esto es lo que hace, poder insertar un valor por teclado. Al insertarlo, procedemos a ejecutarlo.



Resultado de la ejecución del programa:

