

# ABAP/4 Frequently Asked Question

## SAP R/3 Document : FAQ – ABAP/4 Dictionary

### ABAP/4 Dictionary

#### Q. What is a “DATA DICTIONARY”?

A data dictionary is a central source of information for the data in a data management system. Its main function is to support the creation and management of data definitions (or "metadata").

They enable a fast response to the following questions:

- What data is contained in the corporate databases?
- What are the attributes of this data (name, length, format, and so on)?
- What relationships exist between the different data objects?

#### Q. What functions does data dictionary perform?

In a data management system, the principal functions performed by a data dictionary are the following:

- management of data definitions
- provision of information for evaluations
- support for software development
- support for documentation
- ensuring that data definitions are flexible and up-to-date

#### Q. What is Relational Data Model?

A data model is a representation of a section of the real world employing formal structures. The relational data model employs only one such formal structure, referred to as a relation or table. All information on the section of the real world included in the model is shown in the relational model as values in tables.

#### Q. Explain ABAP/4 Dictionary?

The ABAP/4 Dictionary describes the logical structure of the objects used in application development and shows how they are mapped to structures in the underlying relational database. It makes central description of all data used in the system without redundancy possible.

The ABAP/4 Dictionary provides a logical view on the application data and the organization of this data in the underlying database system.

The ABAP/4 Dictionary is an integrated and active data dictionary. Integration in this case means that the ABAP/4 Dictionary is completely embedded in the ABAP/4 Development Workbench. Also, it is active in the run time environment.

#### Q. What information are directly taken from dictionary?

The ABAP/4 Repository Information System provides the user with a wide variety of options for analyzing the use of data and data dependencies. As a result, a high level of transparency of the data structure is obtained for the user. The displayed data also originates from the ABAP/4 Dictionary. A large part of the online help takes its information directly from the ABAP/4 Dictionary. In this way,

the ABAP/4 Dictionary helps to ensure that the online help is up-to-date and to reduce redundancy in the documentation.

The following information is taken directly from the ABAP/4 Dictionary:

- The information on fields displayed with F1 Help
- The possible entries for fields displayed with F4 Help
- The match code and help views search utilities.

### Q. What are the basic objects of ABAP/4 Dictionary?

The Basic elements are.

- Tables
- Data Elements
- Structures
- Domains

### Q. Explain each basic object of ABAP/4 Dictionary?

#### Tables :

In the ABAP/4 Dictionary, tables can be defined in a way, which is not database dependent. For this, fields of the table are defined with their database-independent (external) data types and lengths. In addition, table-specific settings, such as the buffering of the table or indexes for the table, can be defined.

When the table is activated, a physical table definition in the database is added to the table definition stored in the ABAP/4 Dictionary. The database-independent table definition from the ABAP/4 Dictionary is translated into a definition of the relevant database.

The attributes of table fields are distributed over three levels in the ABAP/4 Dictionary (fields, data elements, and domains), each attribute being assigned to only one of these levels, each field must refer to both a data element and a domain. This ensures the consistency of formally and semantically equivalent fields.

The minimum set of fields sufficient to uniquely identify each table line by their values is termed the table **key**.

The **technical settings** allow you to optimize storage space requirements and table access behavior for database tables. Via the technical settings you can determine how the table should be treated when it is created in the database, whether the table is buffered and whether changes to entries should be logged.

The most important parameters in the technical settings are:

**Data class:** The Data class establishes the physical area of the database (table space) in which your table is to be stored.

**Size category:** The Size category allows you to specify estimated space requirements for the table in the database. Efficient access to the data contained in a table can be supported by **Indexes**.

#### Index:

An index could be described as a copy of a database table reduced to specific fields. This copy always exists in sorted form. This sorting enables fast access to the data records of the table by means of binary search, for example. In order that the remaining fields can also be read, that is, those fields not contained

in the index, a pointer to the associated record of the actual table are included in the index.

A distinction is made between **primary and secondary indexes** to a table. The primary index consists of the key fields of the table and a pointer to the non-key-fields of the table. The primary index is generated automatically when a table is created and is created in the database at the same time as the table. It is also possible to define further indexes to a table in the ABAP/4 Dictionary, which are then referred to as secondary indexes. Where there are a number of different indexes to the same table, these are distinguished by means of a three-character **Index ID**.

If the values for the fields of an index uniquely identify a data record of the table, the index is referred to as a **Unique index**.

### Sub structure:

Instead of listing all the fields you wish the table to contain, you can choose to include the fields from another table or structure as a Substructure.

The Modification of SAP standard tables is rendered possible by append structures and customizing includes. These types of modifications are merged with the new standard SAP versions when Release upgrades are performed, ensuring that customer enhancements are not lost.

### Data Element:

A data element (semantic domain) describes the role played by a field in a technical context. Fields of the same semantic meaning refer to the same data element.

A data element carries field information valid for every field that refers to this data element, independently of the table in which a given field occurs. The semantic information includes the field's representation on the screen in the form of field texts, column captions in list outputs of the table contents and the format of the output via parameter IDs. This is also true for the online field documentation (that is, the text displayed in the field help (F1 help) for a screen field is taken from the corresponding data element).

### Domain:

A domain describes the technical characteristics of a table field. It specifies a value range, which describes allowed data values for the fields that refer to this domain.

The value range for a domain is defined by specifying a data type and a length. For example, specifying the data format NUMC and the field length 8 define the number of a person belonging to the university.

The ABAP/4 Dictionary provides two ways of further restricting the value range for a domain within a data type:

- By specifying **fixed values**. For example, the value range for the months in the year can be specified by listing all possible values (January, ..., December).
- By stipulating a **value table**. Only values contained in the relevant field of the value table may be entered in fields referring to this domain. A check is not made on the input mask for a field referring to this domain unless a corresponding foreign key has been defined for this field.

Both fixed values and a value table can be specified for a domain. As a result, fields referring to this domain can accept only those values that exist in both the value table and in the fixed values.

### Structures:

In the ABAP/4 Dictionary the construction of the data produced when calculations are carried out within programs or when data is transferred between programs can also be defined globally. This is achieved by defining a structure. A structure is defined in the ABAP/4 Dictionary like a table and can then be addressed from ABAP/4 programs.

While data in tables is stored permanently in the database, structures contain data only during the run time of a program.

Structures are defined in (almost) the same way as tables. The only difference is that no database table is generated. The same data elements and domains can be used in structures as are used in tables and it is even possible to Include tables. As a result, it is possible to achieve a high level of consistency of data definitions even in complex programs, which link data taken from several places.

### Q. Explain foreign keys in relational data model?

The relational data model contains not only tables, but also relationships between tables. These relationships are defined in the ABAP/4 Dictionary by foreign keys. An important function of foreign keys is to support data integrity in the relational data model. Foreign key fields may assume only those values allowed by the check table, in other words, values occurring in the primary key of the check table.

These also form the basis for defining aggregate objects like views, matchcode etc.

### Q. What are aggregate objects?

Views, match codes and lock objects are called as aggregate objects since they are formed using several related tables.

**Views** a virtual table tailored to the needs of an application is needed, making it possible to directly access specific data.

**Match codes** is a tool to help you search for data records in the system. Match codes are an efficient and user-friendly search aid for cases where the key of a record is unknown.

**Lock objects:** Simultaneous accessing of the same data record by two users in the SAP system is synchronized by a lock mechanism. When dialog transactions are programmed, locks are set and released by calling certain function modules. These function modules are generated automatically from the definition of so-called lock objects in the ABAP/4 Dictionary.

### Q. What are different types of views and explain them ?

**Database views :** normally generated using joining tables. With this an equivalent view is created in the database.

**Projection views:** only for a single table and used to suppress certain fields of the table. Base tables are accessed via R/3 system.

**Help views:** used to display information on online help system. Base tables are accessed via the help processor in the R/3 system.

**Maintenance views:** are used to realize commercially relevant views on data. Base tables are accessed

via special maintenance modules.

Two further types exist for existing views. These view types cannot be used to create new views in Release 3.0.

**Structure views** are used to generate a structure from several logically connected tables. The structure is declared in the ABAP/4 program with the TABLES command and its fields can be accessed in the usual manner. Access is not permitted to these view structures with SQL. Structure views do not differ functionally from pure structures.

**Entity views** are used to represent entity types of the Data Modeler on tables of the ABAP/4 Dictionary. An entity view therefore shows how the attributes of an entity type are implemented in the ABAP/4 Dictionary with table fields. Entity views cannot be used in ABAP/4 programs.

### Q. Define Matchcode ?

**Matchcodes** are defined in two stages in the ABAP/4 Dictionary:

- The relevant tables and fields are stipulated in a Matchcode object. A matchcode object describes the set of all possible search paths for a search term.
- One or more Matchcode ID can be defined for a matchcode object. A Matchcode ID describes a special search path for a search term. The fields or combinations of fields via which the search is to take place are defined in the matchcode ID.

### Q. What are Type Groups?

You use type groups to store user-defined data types or constants in the ABAP/4 Dictionary for cross-program use.

In ABAP/4 program, you declare type groups with the TYPE-POOLS statement.

### Q. What are pooled and cluster tables?

A **table pool (or pool)** is used to combine several logical tables in the ABAP/4 Dictionary.

**Pooled tables** are logical tables that must be assigned to a table pool when they are defined.

Pooled tables can be used to store control data (such as screen sequences or program parameters).

Several pooled tables can be combined in a table pool. The data of these pooled tables are then stored in a common table in the database.

A **table cluster** combines several logical tables in the ABAP/4 Dictionary.

**Cluster tables** are logical tables that must be assigned to a table cluster when they are defined. Cluster tables can be used to store control data (such as screen sequences or program parameters). They can also be used to store temporary data or texts, such as documentation. Several cluster tables can be combined to form a table cluster. The data of the cluster tables is then stored in a single common table cluster in the database.

## Frequently Asked Questions – ABAP/4 Reporting

### 1. What are the two ways in which you can maintain list headers?

You can maintain a list title and up to four column headers for your report list. You do this by choosing *Goto → Text Elements → Title and Headers*.

Another way of doing this is to select *System → Lists → List header* from the created list itself. This has the advantage that the list is displayed on the screen, making it easier for you to position the elements, especially the column headers.

The next time you run the report, the header lines will appear automatically in your list.

### 2. What is the advantage of using text elements?

One of the advantages of using text symbols is that the string is no longer hard-coded in the program. This means that you can maintain the text without having to access the source code. Such a procedure is particularly significant for software which exists in several languages.

### 3. How can create your own data types in an ABAP report?

User defined data types can be created using the `TYPES` statement.

Eg. Types : `mchar(4)` type `c`.

Data : `m` type `mchar`.

### 4. In which system variable is the loop counter stored?

`SY-INDEX`.

### 5. How does the EXIT command works?

You use the `EXIT` statement within a loop structure to leave the current loop.

You use the `EXIT` statement within a subroutine to leave the current subroutine.

If you use the `EXIT` statement outside a loop or a subroutine, it causes the report processing to terminate, and the list is output.

### 6. How does a COLLECT statement work?

`COLLECT` searches in the internal table for an entry, all of whose alphanumeric fields are identical with those of the entry in the work area or header line. If such an entry is found,

`COLLECT` adds all numeric fields from the work area or header line to the corresponding fields in the table entry. Otherwise, the `COLLECT` statement appends the contents of the work area or header line to the end of the table.

### 7. Which system variable stores the record counter of an internal table?

`SY-TABIX`.

### 8. Does the SELECT command carry out any authorization check?

No.

### 9. What does the system variable SY-DBCNT contains?

`SY-DBCNT` contains the number of entries read using a `SELECT` statement.

### 10. What is the function of the environment variable SY-LISEL in interactive reporting?

`SY-LISEL` stores the contents of the line selected. The contents of the variable are available in the secondary screen.

### 11. How is the read/write operation done on an internal table with header line?

When filling an internal table with a header line, you place the data in the header line. Using the

corresponding statements, the contents of the header line is transferred to the table. When reading the internal table, the ABAP/4 runtime system places the contents of the read table line in the header line.

## **12. What is the function of the field SY-LSIND?**

The basic list and up to 20 secondary lists can exist in parallel. The system field SY-LSIND contains the index of the list the report is creating. The system increases the contents of the field by 1 at the beginning of an interactive processing block. You can distinguish the processing for the different list levels by requesting the contents, for example, using a CASE statement. Changing the system field SY-LSIND you can define in which list area the value is output. The index value you choose cannot be greater than the value set automatically by the system. SY-LSIND must be the last statement in the corresponding processing block.

# SAP Document : FAQ – Data Modeler

## DATA MODELER

### ➤ *SAP-SERM*

SAP SERM = (Structured Entity Relationship Model). It supports the user during modeling and when mapping the created model to the ABAP/4 Dictionary.

.

### ➤ *TOP DOWN Model*

First an overall model is created for the area to be modeled. Attributes are defined for the individual entity types. If the technical and semantic characteristics of the attributes are clear, data elements (and domains) can be created or existing ones can be used, based on the data model .

.

### ➤ *BOTTOM UP Model*

The Data Modeler can also be used for (subsequent) modeling of an application, which already exists. In this case, entity types are created for tables and views, which already exist, and the tables and views are assigned to these entity types. The entity types obtain their attributes in this way.

.

### ➤ *How is the link between the Data Modeler and the ABAP/4 Dictionary created?*

Assigning entity types to tables or views creates the link between the Data Modeler and the ABAP/4 Dictionary. New tables or views can be created from the entity type. The attributes of the entity type can be transferred as fields of the table, and the data elements created can be used again. Once this assignment has been made the fields of the view or table are displayed as attributes of the entity type, and are overlaid on the attributes (with the same name) which were previously entered for the entity type. Changes in the assigned Dictionary objects (for example adding, removing or changing fields) therefore take immediate effect, and are visible in the Data Modeler. This ensures the consistency of the data model and Dictionary, in that they access the same information. The relationships of the entity types that were modeled in the data model are stored in the Dictionary as semantic foreign keys in the foreign key relationships. They therefore directly reference the particular modeled reference type or the specified reference type cardinality.



## SAP R/3 Document : FAQ – ABAP/4 Dialog Programming

**1) Why do I need dialog programming in the first place.**

*Ans ) to have your own customized screens*

**2) What does the dialog programming consists / made up of ?**

*Ans ) Screen , Menu , Transaction code , module pool program*

**3) Difference between the normal report / program and Module pool program?**

*Ans) Normal report can be run straight away by F8 – Attributes : I online program*

*Module pool program cannot be run straight away – attributes : M module pool program*

**4) PBO and PAI**

*Ans ) Process before output*

*Process after input*

**5) How can I identify which button is pressed**

*Ans ) “fcode” attributes of the button*

**6) GUI interface to the program**

*Ans ) menu bar*

**7) What does PAI and Pbo contain by default**

*Ans ) PBO - MODULE STATUS\_0100*

*PAI - \*MODULE USER\_COMMAND\_0100*

**8) If we do not have / give menu bar than what will menu bar have by default.**

*Ans ) system , help*

**9) The number of PBO and PAI for a screen.**

*Ans ) one PBO and one PAI*

**10 ) In the menu bar can you associate a function key to a button.**

*Ans ) yes*

# SAP R/3 Document – FAQ SQL / Performance

## 1. What are the different SQL statements used in abap/4?

### a) Open SQL.

They are a set of ABAP/4 commands, which perform operation on database tables. The results of these operations and the associated error messages are independent of database systems used. Open SQL thus offers a unified SQL syntax and semantics of different database systems.

To avoid incompatibilities between different database tables and also to make ABAP programs independent of the database system in use, SAP has created a set of separate SQL statements called Open SQL. Open SQL contains a subset of standard SQL statements as well as some enhancements, which are specific to SAP. Using Open SQL enables you to access any database tables available to the SAP system, regardless of the manufacturer.

There are no DDL and DCL statements in Open SQL.

The open SQL command set include the following command –  
SELECT, INSERT, UPDATE, DELETE, MODIFY, OPEN CURSOR, FETCH, CLOSE  
CURSOR, COMMIT WORK, ROLLBACK WORK.

All SQL Statements used in ABAP other than native SQL.

**NOTE : TO Execute an open SQL command:**

- 1) The addressed database system is supported by SAP.
- 2) The database table is defined in ABAP dictionary.

### b) Native SQL.

These are database specific SQL statements or the ANSI SQL which all RDBMS supports..

#### Syntax

**EXEC SQL [PERFORMING <form>].**

**<Native SQL statement> [;]**

**ENDEXEC.**

Advantage :

- 1) Tables which are not declared in ABAP dictionary can be accessed. ( e.g. TABLES belonging to sys or system user of oracle etc.)
- 2) To use some of the special features supported by the database specific SQL.  
(e.g. Passing hints to Oracle optimizer.)

Disadvantage :.

- 1) No syntax check is performed what ever is written between EXEC & ENDEXEC.
- 2) ABAP program containing database-specific SQL statements will **not** run under different database systems.
- 3) There is no automatic client handling for client dependent tables.
- 4) Care has to be taken during migration to higher versions.

**NOTE : Use of Native SQL is highly discouraged by SAP.**

## 2. How to take care for performance in ABAP development?

### ♦ Keep the data selection small.

- To avoid transporting unnecessary data across the network
- Always use the WHERE clause
- Avoid selecting useless data that you filter out later (using CHECK, for example).

---

**Get more ABAP Hints and Tips:**

<http://www.erpgreat.com/abap.htm>

Page 10 of 16

- Use the indexes of the relevant database tables to make your WHERE clause more efficient, by checking all index fields for equality (EQ, =) and using the AND operator
- Avoid using complex WHERE clauses, since the system has to break them down into several individual statements for the database system
- If possible, avoid using the NOT operator in the WHERE clause, because it is not supported by database indexes; invert the logical expression instead.

♦ **Transport as little data as possible.**

- Transport only the fields of the database table that you really need. If you do not need all of the fields in a table, use a field list in the SELECT clause instead of SELECT \*.
- Use the aggregate functions in the SELECT clause for calculations, instead of transporting large amounts of data and then performing the equivalent calculation.
- Use the UPDATE statement sparingly: Only update the columns that have actually changed, and do not overwrite the entire line.
- Note here that the addition INTO CORRESPONDING FIELDS in the INTO clause of the SELECT statement is only effective for large amounts of data, because the time required to compare the field names is otherwise too great.
- Consider using the DISTINCT option if you are expecting a lot of duplicate table entries.
- **Use fewer database accesses**
- Transfer all of the data at once from the database into internal tables
- Where possible, avoid accessing the same data more than once (for example, by using SELECT before an UPDATE or DELETE statement).
- Avoid nested SELECT loops. Instead, use an internal table and a second SELECT statement with the FOR ALL ENTRIES addition.
- In exceptional cases, you can also select data using a separate cursor.
- **Using Database Buffering**
- Saving database tables in local buffers can save a considerable amount of time. Wherever possible, use buffered data, and only use the BYPASSING BUFFER addition where absolutely necessary.
- Note that the following additions automatically bypass the buffer: DISTINCT, SINGLE FOR UPDATE, and aggregate functions in the SELECT clause.

**TIPS:**

- ❑ You can check the performance of your SQL or ABAP functions by using the ‘**SQL Trace**’ and ‘**Runtime analysis**’ utilities.
- ❑ For tips on how to improve the performance of ABAP tasks, choose transaction **SE30**

**3. What is Runtime analysis and its benefits?**

**Definition:** The Runtime Analysis tool lets you analyze the performance of any transaction or program created within the ABAP Workbench. The Runtime Analysis tool creates lists that reveal expensive statements, summarize table accesses, and show the hierarchy of a program's overall flow. This information enables you to detect and analyze problems resulting from:

- Overuse of or unnecessary calling of modularization units (such as subroutines or function modules) and ABAP statements
- CPU-intensive programming functions
- User-programmed functions that replace existing ABAP statements
- Inefficient and unnecessary database accesses
- find out the database tables accessed by an ABAP program.
- find out resource utilization for the distributed environment of processing for all the 3 tiers of SAP.

#### 4. What is SQL trace tool?

The SQL Trace tool lets you examine the database calls of reports and transactions. This tool shows you:

- the SQL statements that your application uses.
- which values the system uses for specific database accesses and changes
- how the system translates ABAP OPEN SQL commands (such as SELECT) into standard SQL commands
- where your application makes unnecessary database accesses or repeated accesses
- where your application positions COMMIT statements
- what database accesses or changes occur in the update section of your application
- which index and with what cost it is used for data retrieval.

**NOTE :** Only one user can use the trace tool at a time. So, it is very important to switch off the tool as soon as the execution is over.

#### 5. What is Explain SQL function?

The *Explain SQL* function provides you with an analysis of a database's strategy for accessing any ABAP Dictionary object. You can use this analysis to identify the indexes used for database access.

#### 6. When to buffer a table?

A table should be buffered when it is

- Rather small in size
- Accessed mostly for read purposes
- Changed very infrequently Tables which are very good candidate for buffering:
- Control tables / customizing tables
- "SMALL" master data tables

#### 7. Which SQL statement bypasses the table buffer?

Following are the SQL statements which bypasses table buffer:

- SELECT ----- BYPASSING BUFFER.
- SELECT from database views (projection views are OK)
- SELECT ----- DISTINCT -----.
- SELECT ----- COUNT, SUM, AVG, MIN, MAX.
- SELECT ----- ORDER BY (other than the primary key)
- SELECT ----- FOR UPDATE. ( EXEC ----- ENDEXEC)
- Native SQL statements.

**NOTE :** Avoid this statements when working with buffered tables

#### 8. What is an expensive select statement?

- Long response time
- Many data blocks are scanned to find the selected records.
- Long DB request time caused by too many buffer gets.

#### 9. Explain the open SQL statement modify?

**This is used for Adding or changing lines in DB tables.**

To insert a line into a database table, regardless of whether the primary key of this line already exists, you use the MODIFY statement.

There are two possibilities:

- If the database table contains no line with the same primary key as the line to be inserted, MODIFY works like INSERT, i.e. the line is added.
- If the database already contains a line with the same primary key as the line to be inserted, MODIFY works like UPDATE, i.e. the line is changed.

Note : For performance reasons, you should use MODIFY only if you cannot distinguish between these two options in your ABAP program.

#### *10 . How automatic client handling is switched off in Open SQL?*

The CLIENT SPECIFIED option switches off automatic client handling. You can then specify the client in a WHERE condition, and fill the client field in table work areas.

# SAP Document : FAQ – WorkBench Organizer

## ➤ *Workbench Organizer*

The Workbench Organizer in the ABAP/4 Development Workbench provides functions for organizing development projects. It is fully integrated into the customizing tools as the Customizing Organizer and is designed to support projects of all sizes, whether they are carried out centrally or in a distributed environment.

## ➤ *Request Overview*

If you select the Display function in the initial screen of the Workbench Organizer, you are presented with a hierarchical list of change requests. The list is arranged according to the chosen selection criteria. Double-click on + or - to access the next level of information or to compress all subordinate nodes.

## ➤ *Difference between transaction codes se09 and se10.*

SE09 is the transaction code for Workbench Organizer's change request for development objects. SE10 includes SE09 and customizing requests.

## ➤ *Table in which the entry for the Development class is made.*

table TADIR

## ➤ *Naming Convention of the Development Class:*

1. Development classes beginning with A-S or U-X

These classes are reserved for SAP standard objects. Customer objects may not be created in them. Changes to objects in these classes are recorded in the Workbench Organizer and can be transported, if the system is set appropriately. All these classes belong to the "SAP" transport layer (see the section Tables for R/3 System Configuration).

2. Development classes beginning with Y or Z

Objects belonging to the customer can be created in these classes. Changes to the objects are recorded in the Workbench Organizer and can be transported, if the system is set appropriately.

3. Development classes beginning with T (private test classes)

When you create such a class you can specify whether or not it should be subject to the control of the Workbench Organizer. If this is the case, objects that you modify will be included in local requests of the Workbench Organizer that are not intended for transport.

The class is not assigned to a transport layer.

The objects may only be transported to other SAP systems in special transport requests (transports of copies or originals).

Newly installed systems always include a private test class called "TEST", that is not linked to the Workbench Organizer.

4. Development classes beginning with \$ (local classes)

Changes to objects of this class are not recorded in the Workbench Organizer. The class does not belong to a transport layer and the objects cannot be transported.

In newly installed systems one local class, called '\$TMP', is available.

## ➤ *What do you mean by Customizing Requests?*

Customizing requests (change requests of the category CUST) record client-specific customizing settings within a client. A request/task of the category CUST only contains client-specific customizing objects of the category CUST that originate from exactly one client (source client).

➤ *Change Requests?*

Requests/tasks of the category SYST contain objects that also belong to the category SYST.

These are ABAP/4 Repository objects and customizing objects intended for all clients.

Requests/tasks of the type SYST can also contain objects belonging to the category CUST, i.e. they allow a combination of both categories. However, as above, all the objects may only originate from one client (the source client).

➤ *Which Client is authorized to create the workbench Organizer configuration?*

000 client.

# Resource Guide

## Recommended Reading

### **Practice ABAP From Your Home PC**

Minisap contains only SAP BASIS where you can learn and practice ABAP at home:

<http://www.erpgreat.com/mini-sap.htm>

### **More than 100 ABAP Interview Faq's**

Large list of useful abap interview questions.

<http://www.erpgreat.com/abap/more-than-100-abap-interview-faqs.htm>

### **Example of a Simple ALV Grid Report**

A sample ALV Grid program for your reference.

<http://www.erpgreat.com/abap/example-of-a-simple-alv-grid-report.htm>

### **mySAP Certification - Criteria For Application**

The guidelines: Criteria to applying for a SAP Certification :

<http://www.erpgreat.com/general/mysap-certification.htm>

### **ABAP and Functional Modules Table**

As an Abaper, you need to know all the available functional modules tables so that you know where to extract them for your users and functional consultant.

<http://www.erpgreat.com/general/sap-r3-tables.htm>