جامعة الملك فهد للبترول والمعادن
King Fahd University of Petroleum & Minerals

College of Computer Sciences and Engineering [CCSE]

Department of Information and Computer Sciences

SWE 399 – Summer Training Report

[Term 093]

Shaeq Pervez Khan, 200792450

Summer Training Coordinator:
Dr. Husni Muhtaseb, ICS Dept.

Training Date:
Start: 28th June 2011
End: 25th August 2011

Wipro Ltd.
Wipro KODC,
Info Park Special Economic Zone
Kusumagiri PO, Kakkanad
Kochi – 682 030, India.
www.wipro.com

Department – SAP ENT Practice
[ABAP Training & project work]

Direct Supervisor:
Mr. Praveen Sharma, PMP, CISA
Delivery Manager, Project Manager
Email: praveen.sharma@wipro.com
Phone no: 0091 989 5315140

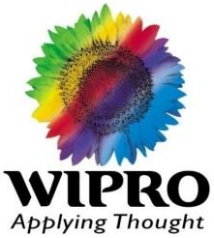

Table of Contents

Acknowledgement

I would like to express my sincere appreciation to my technical supervisor Mr. Neeraj Chhibbar for the knowledge he has passed on to me and valuable feedback in preparation of this document. I would like to thank Mr. Praveen Sharma, Mr. Praveen Alex, Mr. Abhishek Nayar and Mr. Iftekhar Nadeem for providing me with this opportunity to work with the Wipro family and gain invaluable experience and knowledge.

A Big thank you to Aisha Mehta, Anupam Sharma, Fahad Wasim, Nageshwar KN Rao, Shama Farha, Somprasad Nimmalla, Sujoy Roy, Swapnil Velhal, Toufeeq Emraan, Tushar Sharma and Vignesh Pillai for being so supportive and making me feel at home in Kochi.

Thank you Baba for staying strong and giving me the courage to take this opportunity.

WIPRO
Applying Thought

Aug 24th 2011

Student Name: Mr Khan, Shaeq Pervez
University: King Fahad University of Petroleum and Minerals
Course: BS Software Engineering 2008-2012

Dear Mr. Shaeq,

Congratulations!!!

We would like to convey our best wishes for successfully completing your summer training at Wipro Cochin ODC and completing your project assignment in SAP ABAP from June 27th 2011 to Aug 24th 2011. During this period following topics on SAP ABAP were covered:-

1. SAP ABAP Workbench Concept
2. Database Concept
3. ABAP Dictionary
4. List Reports
5. ABAP Module Pool

We appreciate your dedication, commitment, enthusiasm and initiative in taking the tasks to completion during the program. All the best for you future and career aspirations

With Best Wishes,
For Wipro Ltd

Praveen Sharma
Delivery Manager

WIPRO LIMITED Kochi

Wipro Limited, Wipro Campus, Info Park SEZ, Kusumagiri (PO), Kakkanad, Kochi-682030, Tel:0484-3054949, Fax: 0484- 3054224, www.wipro.com.

Introduction

SAP was founded in 1972 as System Analysis and Program Development by 5 former IBM engineers. The acronym was later changed to Systems, Applications and Products in Data Processing. SAP is used for developing customized applications and modifying existing standard applications.

SAP is the world's largest business company. It operates in four geographic regions - EMEA (Europe, Middle East, Africa), AMERICA (United States and Canada), LAC (Latin America and Caribbean), and APJ (Asia Pacific and Japan), which represents Japan, Australia, India, and parts of Asia.

SAP Branching

SAP has two branches –

a. Technical Area
[Coding + Configuration]

b. Functional Area
[Configuration + Exploration]

I decided to do my training in the Technical Area. It is further divided into the following –

i. ABAP – Advanced Business Application Programming

ABAP is a High Level Language [HLL] and is one of the many application-specific fourth-generation languages. *ABAP* was one of the first languages to include the concept of *Logical Databases* (LDBs), which provides a high level of abstraction from the basic database level(s).

The *ABAP* programming language was originally used by developers to develop the SAP R/3 platform. It was also intended to be used by SAP customers to enhance SAP applications.

Customers can develop custom reports and interfaces with *ABAP* programming. The syntax of *ABAP* is somewhat similar to COBOL.

ii. Basis

Basis is a business application software integrated solution. Simply, *Basis* is the administration of the SAP system. The *Basis* administrator is an SAP professional who is responsible for configuring the SAP environment, including the GUI screens and SAP application servers.

iii. BI - Business Intelligence [presently known as Data Warehousing (DW)]

It refers to computer-based techniques used in identifying, extracting and analyzing business data, such as sales revenue by products and/or departments. *BI* technologies provide historical, current and predictive views of business operations.

iv. XI – Exchange Infrastructure [presently known as Process Integration (PI)]

SAP Exchange Infrastructure (*SAP XI*) enables you to implement cross-system processes. It enables you to connect systems from different vendors (non-SAP and SAP) in different versions and implemented in different programming languages.

Amongst these categories, I successfully completed my training in ABAP on the topics – Reports and Module Pool.

REPORT

There are 2 types of reports:

1. Classic Reports
2. Interactive Reports

Classic reports do not allow interaction by the user; therefore, the basic list contains extensive information that the user must often sort through to find the relevant data.

Interactive reports allow interaction by the user; therefore, the user can produce secondary, detailed lists off of the basic list by choosing the relevant data and requesting more information.

Reports are used when data from a number of tables have to be selected and processed before presenting to the user. It is an event driven programming.

MODULE POOL

Module Pool programs are a collection of screens, their Flow Logic, and the code within the main ABAP program. The term *"screen"* refers to the actual, physical image that the user sees.

The term *"flow logic"* refers to the code behind the screens (i.e., the logic that initializes screens and responds to a user's requests on the screens). Each screen has its own Flow Logic.

- The *Screen Painter* is used to maintain all components of the screens.

- The *ABAP Editor* is used to maintain the main ABAP program. This program consists of data declarations and modules. The ABAP modules contain the main processing logic of the online program and are "called" from within the Flow Logic of the screens.

- The *Menu Painter* is used to maintain the graphical user interface (GUI) which consists of the function key assignments, standard toolbar, application toolbar, menu bar and title bar.

- The *ABAP Dictionary* is used to link dictionary field definitions to screen fields as well as to create program work areas (with the TABLES statement).

About Wipro

Establishment of Wipro

Wipro was founded in 1945 by M.H.Hasham Premji in Amalner, Maharashtra, producing sunflower vanaspati oil and soaps. At that time, the company was called *Western India Vegetable Products Limited* (later abbreviated down to Wipro). The company logo still contains a sunflower to reflect their original business.

During 1970s and 1980s it shifted its focus and began to look into business opportunities in IT and computing industry which was at nascent stages in India at that time.

Wipro was the first company which marketed the first indigenous homemade PC from India in 1975. In the early 1980s, Wipro began selling through a dealer network and began assembling products made by well-known companies such as Canon, Cisco Systems, Epson, Hewlett-Packard and Sun.

By 2000, Wipro Technologies emerged as the largest publicly listed software exporter in India and the first software services provider to be assessed at SEI Level 5 in the world.

Location of the Company

Wipro has a worldwide presence in India, Middle East, Japan, France, Germany, China, Australia and New Zealand.

WIPRO VISION

- Contribute for global e-society, where a wide range of information is being exchanged beyond time and space over global networks, which breaks down the boundaries among countries, regions and cultures, allowing individuals to take part in various social activities in an impartial, secure way.
- Continuous effort to enhance people's lifestyle and quality by means of developing new technology in wireless communication.

WIPRO MISSION

Our mission is to be a RF System Solution Provider, through its innovative research and design works for a new world of broadband wireless communications.

WIPRO GOALS

- To support customers who rely on our ability as an advanced RF System Solution Provider.
- To build up core competencies through collaboration with technological partners.
- To contribute to the Ubiquitous Networking Society by providing chip level RF system solutions.

WIPRO INDUSTRIES

Aerospace, Automotive, Banking, Business and Consumer Service, Communication Service Providers, Computer Peripherals, Computer Software, Computing, Consumer Electronics, Consumer Packaged Goods, Energy, Government, Healthcare, High-Tech, Hospitality and Leisure, Insurance, Manufacturing, Media, Medical Devices, Mobile Devices, Natural Resources, Pharmaceutical and Life Sciences, Public Infrastructure, Retail, Securities and Capital Markets, Semiconductors, Storage, Telecom, Telecom Equipment, Transportation, Utilities.

WIPRO SERVICES

- *Consulting* – think ahead with the right strategic insights.
- *Analytics and Information Management* - A comprehensive Information Strategy when applied in an organization.
- *Business Application Services* - Enables customers create successful and adaptive businesses.
- *Infrastructure Management Services* - Step up the pace in technology innovation.
- *Business Process Outsourcing* - Bring flexibility and cost effectiveness into business.
- *Product Engineering Services* - Define, manage, study and execute ideas that matter.
- *Enterprise Technology Integration* - Adapt to the evolving technology landscape.
- *Total Outsourcing* - Achieve maximum value through managed requirements.

WIPRO VALUES

Outstanding, Teamwork, Challenge, Spirit

THE SPIRIT OF WIPRO

Intensity to Win

- Make customers successful
- Team, Innovate, Excel

Act with Sensitivity

- Respect for the individual
- Thoughtful and responsible

Unyielding Integrity

- Delivering on commitments
- Honesty and Fairness in action

Diversification of Wipro

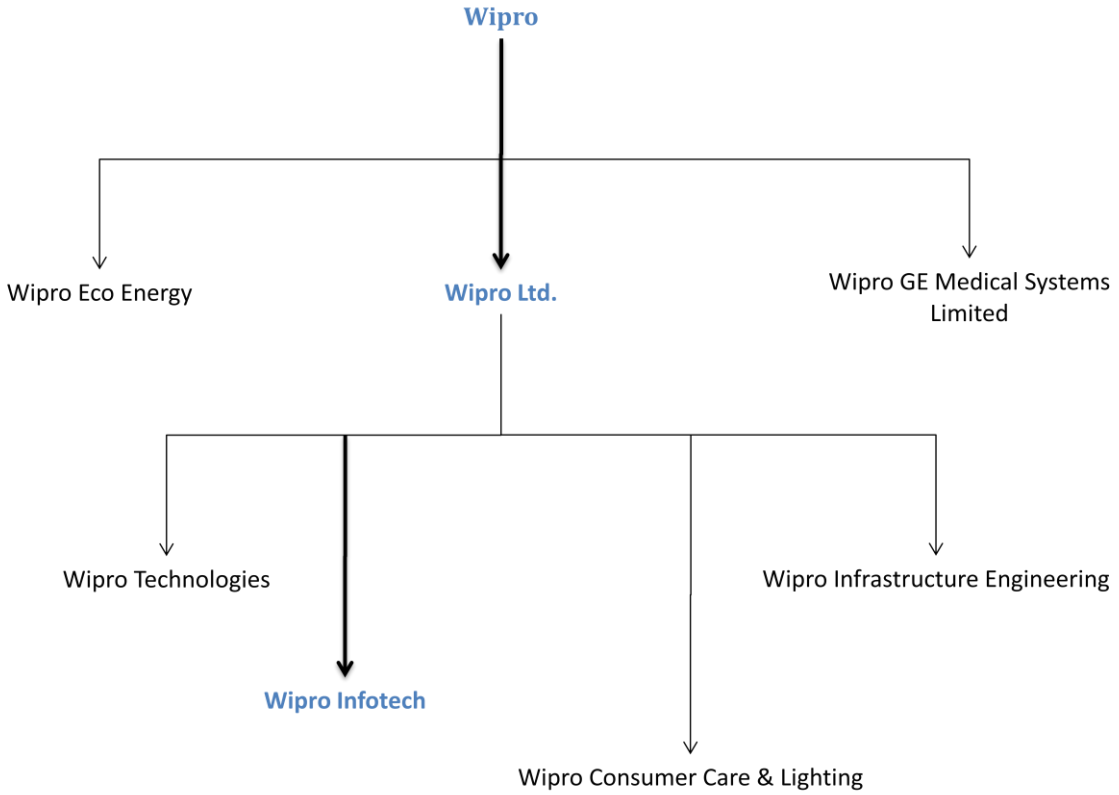

Figure 1 – Wipro diversification

WIPRO ECOENERGY

Wipro EcoEnergy, the renewable energy services arm of Wipro, is a one-stop shop for all the renewable and alternative needs of an organization. The scope of work provides the entire range of sustainable and energy efficient solutions such as,

- Customized clean-energy solutions for institutional clients
- Energy Efficiency (reduce) and Renewable Energy (replace)
- Consulting, implementation and managed services

They work on the industrial scale; the technologies are proven, have direct customer relevance, and are commercially viable. They are technology agnostic because they work on best technologies and recommend the same based on the clients unique environment. Wipro EcoEnergy provides clean and sustainable energy solutions for all kinds of spaces, from factories to institutions and from offices to homes.

WIPRO GE MEDICAL SYSTEMS LIMITED

Wipro GE Medical Systems Limited is focused on delivering advanced solutions for significant challenges faced by healthcare organizations today. Wipro GE Healthcare, a joint venture between

Wipro and GE, is part of GE Healthcare South Asia and caters to customer and patient needs with a commitment to uncompromising quality.

They successfully manage a continuum of clinical information across the entire enterprise and our highly energized team provides superior customer and patient satisfaction, maximizing customer productivity with Six Sigma quality and uncompromising integrity.

WIPRO TECHNOLOGIES

The majority of Wipro Technologies business comes from the US, followed by Europe. The rest of the world contributes only marginally to its top line. Founded in 2002, Wipro Technologies has operations in Delhi, Pune, Kolkata, Chennai, Mumbai, Hyderabad, Navi-Mumbai (Belapur) Greater Noida, Mysore and Kochi in India.

It also has offices in Shanghai and Cebu in Asia and Curitiba in Brazil and Wroclaw in Poland. It has 44 clients in segments such as banking & capital markets, insurance, travel & hospitality, hi-tech manufacturing, telecom and healthcare.

WIPRO INFOTECH

Wipro Infotech is a leading manufacturer of computer hardware and provider of systems integration services in India and the Middle East region. The company's product portfolio includes desktop and notebook PCs, servers, data storage systems, and supercomputers. Its menu of technology services includes application development, data hosting, and technology procurement consulting.

As an IT division of Bangalore-based technology services and consumer products conglomerate Wipro Limited, Wipro Infotech maintains offices across India. It also has operations in Australia and Egypt, as well as in Saudi Arabia through a joint venture with DAR Al Riyadh Group.

WIPRO CONSUMER CARE AND LIGHTING

Wipro Consumer Care and Lighting (WCCLG), a business unit of Wipro Limited, started with vegetable oil production in 1947 and has since come a long way and established a profitable presence in the branded retail market.

With a vast plethora of products spanning soaps, baby care products, health and wellness, Wipro's products have touched the lives of millions of consumers across India and global markets.

It is also a leader in institutional lighting in specified segments like software, pharma and retail. Wipro Consumer Care and Lighting has been one of the fastest growing FMCG companies, both organically and through acquisitions.

WIPRO INFRASTRUCTURE ENGINEERING

Wipro Infrastructure Engineering, a division of Wipro Limited, delivers precision-engineered, world-class hydraulic cylinders, components and solutions, and truck hydraulic components to OEMs globally in the infrastructure and related industries. It represents the Kayaba, Kawasaki, Sun Hydraulics and Teijin

Seiki range of hydraulic products in India.

With state-of-the-art manufacturing facilities and extensive product development and testing facilities, Wipro Infrastructure Engineering has emerged as a leader in the hydraulic cylinders and truck tipping systems market in India. Wipro has recently entered water treatment business and provides ultra-pure water treatment systems and solutions for various industries.

Organizational Structure

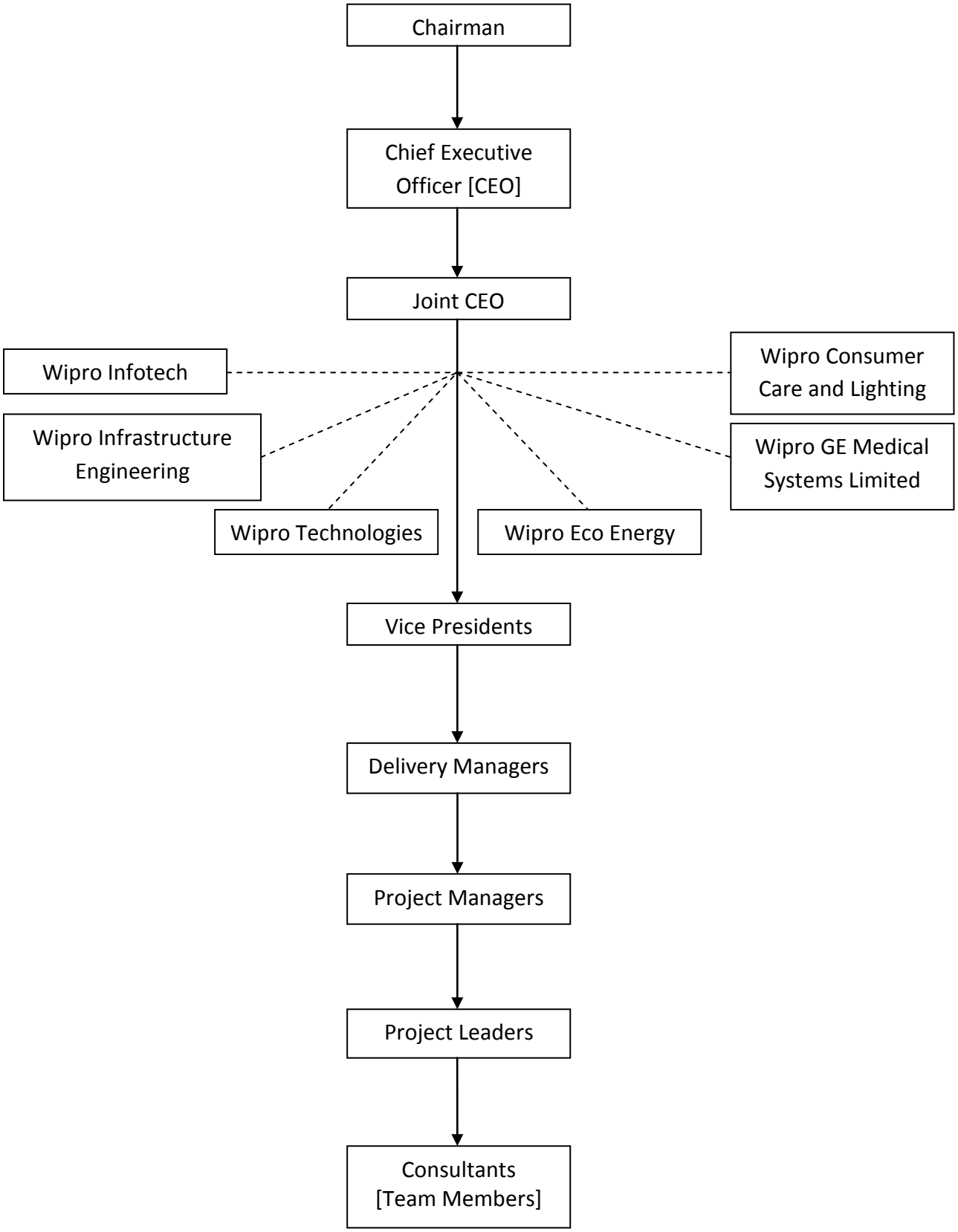

Figure 2 – Organizational Structure

Work Experience

Project 1 – Dynamic Stock Status Report

DEPARTMENT

SAP ENT Practice

The SAP ENT Practice department is situated inside the Wipro campus in tower S2 on the second floor. A section of the floor consists of a huge room divided into several work spaces provided with PC's for each employee.

ABSTRACT

This module is an interactive report type where a user enters a required set of values on the input screen and requests data from different standard tables in the ABAP Workbench. A list is generated on another screen and the user who placed the request can view this screen. As per requirements, certain list areas are made clickable. This list needs to be linked to an available transaction that will pull values from the generated list according to what value is clicked on by the user.

REQUIREMENTS

Selection Screen				Standard Table	Field
Material Code		To		MARA	MATNR
Plant		To		MARC	WERKS
Storage Location		To		MARD	LGORT
Batch		To		MCHB	CHARG
Valuation Type		To		MBEW	BWTAR

- User should be allowed to enter data in any of the fields
- Pass the combination of MATNR + WERKS + LGORT to get the CHARG
- Output should be according to the field specified by the user
- Make 'Material Code' as mandatory
- If we are passing more than one field i.e. Plant, Batch and Material Code, records satisfying all details should come in the list.

Output list should contain the following fields

Material	Material Type
Material Description	External Material Group
Old Material Code	Storage Location
UOM	Valuation Type
Plant	Batch
Material Group	

Make 'Material Code' as a hotspot, clicking on it should take you to a transaction 'MMBE' (Stock Overview) bypassing selection screen for corresponding values of Plant, Batch, Storage Location and Material in that row.

Detailed Design and Implementation

This was submitted as a deliverable to the company and is attached in this report as 'Appendix E'.

Hardware used

Hardware manufactured and supported by Wipro Technologies
Intel(R) Pentium(R) D CPU 2.80GHz
2.79 GHz, 0.99 GB of RAM

Software Packages

ERP: SAP R/3 ECC 6.0
The SAP ERP Central Component 6.0 was launched in 2006 to replace the R/3 enterprise. The purpose of positioning it as ECC is to enable SAP to build and develop an environment of other products that can function upon the foundation of the central component.

People I dealt with

Mr. Praveen Sharma
[Project Manager]

Mr. Neeraj Chhibbar
[Technical Supervisor]

Mr. Somprasad Nimmalla
[SAP ABAP Consultant]

Mr. Nageshwar KN Rao
[SAP ABAP Consultant]

Role in the project

This module was a part of an un-named project [project name and details were not disclosed to me because of company privacy policies] and was primarily a part of a group of modules to be developed by my technical supervisor Mr. Neeraj Chhibbar.

This module development was under MM [Material Management]. The module titled 'Dynamic Stock Status Report' had clearly stated requirements and employed the usage of topics I had just finished learning in my training period of 3 weeks.

My technical supervisor thought it would be apt to consider me as a shadow consultant [trainees working under Technical Consultants or Project Team Members] for a brief period and work in parallel with him on this module.

Learning Experience

I received suggestions and ideas on how to approach in developing this module from my colleagues. I developed the entire module in a period of one and half weeks including the time taken for R&D of new topics and coding, documenting and testing the module.

Once I finished the code I was asked by my supervisor to debug the code using the ABAP Workbench built in debugger and understand the flow and logic of it. This step cleared a lot of my concepts over the flow logic of my own code.

On completion, I was required to give an open demonstration of my work. The presentation included explaining theoretical concepts used in making the module and a one on one question and answer session over how I completed everything over the given period of time.

This involvement as a shadow consultant gave me a good insight as to how it feels to work after training

completes for a proper employee. It gave me a good sense of expectations kept from employees on this post and also an insight into the work ethics of Wipro employees. Some days we would work for up to 14 hours a day and other days would be the usual 9 and half hours per day.

This module development really taught me how to push myself onto a decided project time frame and do useful research over the internet on specific ABAP topics. Sometimes I would not find anything related to what I wanted to learn so this is where my seniors stepped in and explained me things from top to bottom. It was impressive to see their domain knowledge because of their years of expertise.

Future Aspects

My work on this module has made me confident about my knowledge about the following topics –

- ABAP Workbench
- Basic ABAP Language Elements
- Data Retrieval
- Subprograms in ABAP
- ABAP Events
- Classic List Processing
- Function Groups and Function Modules
- ABAP OpenSQL
- Input Checks

All of these happen to the most basic knowledge required for an ABAPer [technical consultants as referred to in the software industry]. And since my knowledge and confidence has developed over from just theoretical and practice exercises to real module development, as a student this is a huge plus point for me in terms of acquiring a job based on industry knowledge.

Besides the topics mentioned above, extensive R&D over the period of time also helped me learn a lot of non-textbook knowledge which are considered to be the 'tricks of the trade'.

Limitations

Problems faced?

The only genuine problem I faced throughout my stay at Wipro was the amount of individual research that was required to be done. It was not really a problem but more like something that was different from what I've been doing at university. I had to rely completely on online documentation and blogs to find answers to my problems. If all options were exhausted, then I got the chance to ask a senior about it.

What was missing?

In my opinion, everything I did was a new experience for me. So I cannot make a statement on what was missing as a whole because I don't have a baseline to compare it to. So I would say that as my first experience of working in a company along with learning a completely new technology, I think I am quite satisfied with the amount and quality of work I have done and the appreciation I have achieved from my seniors at the company.

How could you have done your work better?

I believe there is no other way in which I could have done my assignments and work better. The amount of work load I had was just enough in the beginning and then it kept on growing as time progressed once I got familiar with the technology to keep me alert and working as much as I was required to. The balance of development and research work created by my supervisor was close to perfect.

Also the assistance I received from my colleagues at work regarding the quality of the deliverables according to Wipro standards was very helpful. In my opinion, everything I worked on was very well executed because of the assistance I received along the way to do things the right way.

How could you have gained more experience?

The only way I could have gained more experience as an ABAPer would be spending more time in training and working further more as a shadow consultant on bigger and more complex modules. This way I would be able to move to the next step, knowledge and experience wise.

There are no shortcuts to being an expert in the technical field, so the more time and effort you put in, the better you get at being a developer. The next level for a trainee is working on a live object [objects that are a part of an ongoing project and have dependencies on the project schedule] which is considered to be a good addendum to personal experience.

Project 2 – Flight and Booking Information System

DEPARTMENT

SAP ENT Practice

The SAP ENT Practice department is situated inside the Wipro campus in tower S2 on the second floor. A section of the floor consists of a huge room divided into several work spaces provided with PC's for each employee.

ABSTRACT

This is a module pool where a user enters a value on the input screen and requests data from a user created table in the ABAP Workbench. A table control is populated based on the input value. As per requirements, user can insert, delete and modify contents of the table. To commit all changes made, user has to save. Once saved, all the changes made to the table will reflect on the database.

REQUIREMENTS

Create a *Flight and Booking Information System* using module pool where the user enters the 'Carrier ID' and can view the data from the table SFLIGHT in a table control. If the user clicks on 'Edit' button, the entire table control becomes enabled and user can make changes to entries and save it. If the user selects a row and clicks on 'Insert' button, an initial new line is inserted above the selected line. Also if the user selects a row and clicks on 'Delete', the row is deleted in the table control. The user can save these changes made in the table control to the database by clicking on the 'Save' button. If the user selects a line and clicks on 'View Bookings' then the flow goes to another screen where a list of all booking details from standard table SBOOK for the selected flight is displayed as a list according to the 'Carrier ID' and 'Connection ID'. The user can traverse between the 2 screens freely and perform as many changes and generate as many lists in the same session.

Detailed Design and Implementation

This was submitted as a deliverable to the company and is attached in this report as 'Appendix E'.

Hardware used

Hardware manufactured and supported by Wipro Technologies
Intel(R) Pentium(R) D CPU 2.80GHz
2.79 GHz, 0.99 GB of RAM

Software Packages

ERP: SAP R/3 ECC 6.0
The SAP ERP Central Component 6.0 was launched in 2006 to replace the R/3 enterprise. The purpose of positioning it as ECC is to enable SAP to build and develop an environment of other products that can function upon the foundation of the central component.

People I dealt with

Mr. Praveen Sharma
[Project Manager]

Mr. Abhishek Nayar
[Technical Supervisor]

Mr. Vivek Viswanathan
[Development Consultant]

Mr. Somprasad Nimmalla
[SAP ABAP Consultant]

Mr. Fahad Wasim
[SAP ABAP Consultant]

Role in the project

This module was an independent development assignment under the guidance of my technical supervisor. To test my understanding of the topic 'Module Pool', I was given a requirement statement and asked to develop a module based on it.

To develop this module titled 'Flight and Booking Information System', I had to first write down a simple requirement document clearly stating the scope of the module as per my understanding and then get it verified by my supervisor. This was to make sure that the scope covers the usage of topics I had just finished learning in my training period of 3 weeks.

Once this document was finalized, I proceeded with coding. Since the module development was on a higher difficulty level than the previous one, I was teamed up with Mr. Somprasad who was responsible to evaluate my progress every day and assist me in case I had difficulty understanding the concepts.

Learning Experience

I kept getting suggestions and ideas on how to approach in developing this module from my colleagues. I developed the entire module in a period of one week including the time taken for R&D of new topics and coding, documenting and testing the module.

Once I finished the code I was asked by my supervisor to debug the code using the ABAP Workbench built in debugger and understand the flow and logic of it. This step cleared a lot of my concepts over the flow logic of my own code.

On completion, I was required to give an open demonstration of my work. The presentation included explaining theoretical concepts used in

making the module and one on one question and answer session over how I completed everything over the given period of time.

The teaming up with an experienced programmer and giving daily verbal reports to him actually helped me self-evaluate myself as to where my understanding of certain concepts is lacking and on what topics I should concentrate more. I also enjoyed my conversations with him over walks around the Wipro campus where he would give me ideas on not just the logic for this module. Also I told him about my senior project I was doing at university and he was happy to know about it and gave me lot of suggestions and ideas as to how I can improve the functionalities without going out of the scope.

It gave me a good sense of expectations kept from employees on this post and also an insight into the work ethics of Wipro employees. Some days we would work for up to 14 hours a day and other days would be the usual 9 and half hours per day.

This module development really taught me how to do useful research over the internet on specific ABAP topics. Sometimes I would not find anything related to what I wanted to learn so this is where my seniors stepped in and explained me things.

FUTURE ASPECTS

My work on this module has made me confident about my knowledge about the following topics –

- Dynamic Programming
- Views and Maintenance Dialog
- Screen Programming
- Input/Output elements
- Subscreens

All of these happen to the most basic knowledge required for an ABAPer [technical consultants as referred to in the software industry]. And since my knowledge and confidence has developed over from just theoretical and practice exercises to real module development, as a student this is a huge plus point for me in terms of acquiring a job based on industry knowledge.

Besides the topics mentioned above, extensive R&D over the period of time also helped me learn a lot of non-textbook knowledge which are considered to be the 'tricks of the trade'.

LIMITATIONS

Problems faced?

The only genuine problem I faced throughout my stay at Wipro was the amount of individual research that was required to be done. It was not really a problem but more like something that was different from what I've been doing at university. I had to rely completely on online documentation and blogs to find answers to my problems. If all options were exhausted, then I got the chance to ask a senior about it.

What was missing?

In my opinion, everything I did was a new experience for me. So I cannot make a statement on what was missing as a whole because I don't have a baseline to compare it to. So I would say that as my first experience of working in a company along with

learning a completely new technology, I think I am quite satisfied with the amount and quality of work I have done and the appreciation I have achieved from my seniors at the company.

How could you have done your work better?

I believe there is no other way in which I could have done my assignments and work better. The amount of work load I had was just enough in the beginning and then it kept on growing as time progressed once I got familiar with the technology to keep me alert and working as much as I was required to. The balance of development and research work created by my supervisor was close to perfect.

Also the assistance I received from my colleagues at work regarding the quality of the deliverables according to Wipro standards was very helpful. In my opinion, everything I worked on was very well executed because of the assistance I received along the way to do things the right way.

How could you have gained more experience?

The only way I could have gained more experience as an ABAPer would be spending more time in training and working further more as a shadow consultant on bigger and more complex modules. This way I would be able to move to the next step, knowledge and experience wise.

There are no shortcuts to being an expert in the technical field, so the more time and effort you put in, the better you get at being a developer. The next level for a trainee is working on a live object [objects that are a part of an ongoing project and have dependencies on the project schedule] which is considered to be a good addendum to personal experience.

SUMMARY

WORK EXPERIENCE

It was a pleasure to be a part of the Wipro Infotech family for a brief period of two months. I was never made to feel like a person who has come from outside with no experience for a limited period.

Everyone was extremely supportive and guided me on how best to utilize my two months in learning as much as I can. I never felt tired at the office or ever felt the need to complain about staying in late to finish work.

I liked the work environment so much that I voluntarily went in to work six days a week for the entire training period of eight weeks. I enjoyed every second I spent at the office and with colleagues. I was fortunate enough to make a lot of friends. I enjoyed a healthy and mutually satisfactory relationship with all my superiors.

And most of all the technical expertise I have acquired being a part of this company has boosted my confidence multiple folds and I feel confident enough that I can grasp any new technology in a matter of weeks or continue working as an enthusiastic and successful ABAPer.

Courses heavily used?

ICS 324 – Database Systems

Helped me to easily finish off with the topics under OpenSQL which is the database language used in ABAP Workbench.

SWE 312 – User Interface Design

Helped me to design output interfaces of higher quality in comparison to other trainees.

ICS253 – Discrete Structures I

Helped me create logic for database queries and also design simple algorithms used in my modules.

ENGL 214 – Academic and Professional Communication

Used when I had to submit deliverables, write emails to communicate with employees at the office.

SWE 215 – Software Requirements Engineering

Used when I had to create a requirements document for the second module.

SWE 205 – Introduction to Software Engineering

This course gave me a basic idea of the life cycle of software under production and I was already familiar with the technical terms used in the industry.

Courses you wish you had taken before summer training?

ICS 353 – Design and Analysis of Algorithms

It would have been an advantage and more efficient if I was quicker at creating the algorithms and determining logic flows for my modules.

Courses you think the department should introduce?

Considering the amount of demand for ERP technical experts in the industry, it would a good advantage for a KFUPM student to be familiar with the different ERP systems in the market and their basic workings with some coding too.

Recommendation to pass on to next group of summer trainees?

According to the current situation in KSA software market, if you really want to learn proper development work, gain technical knowledge and be around experts who will always challenge you, then you should consider other options than the KSA market and working in KSA as a technical developer should be your last option. Do not limit your choices because the "real technical expertise" is not in KSA.

What recommendation you want to pass to the department and coordinator?

I have two points I would like to make to the department and coordinator –

1. Please do not underestimate your students over their capabilities and responsibility. You have taught us and trained us well, so please put in some faith in us. Please do not force your students to work exclusively in KSA, let them explore options outside and support them over their choices.
2. The concept of being paid as a student trainee is something that should not be a necessity for accepting the company by the department. Yes it is important that the company treats us as a proper employee but this treatment as an employee entirely depends on the attitude of the student. Not all students are the same, so to generalize all them being irresponsible at work if they are not being paid is unfair.

Resources

TAW ABAP Workbench Fundamentals 10_1*

TAW ABAP Workbench Fundamentals 10_2*

TAW ABAP Workbench Fundamentals 12_1*

http://help.sap.com

http://www.saptechnical.com

http://www.sapdev.co.uk

Appendix A – Weekly breakdown of activities

	ACTIVITY
WEEK 1	○ Introduction to Wipro company rules and regulations for trainees. ○ Campus tour and orientation of services offered to trainees. ○ Trainee's group meeting and brief one on one interview session with would be technical advisor to assess skill level and competency. ○ Introduction and overview of course material and expectations from trainees by the end of training session. ○ Developing acquaintance with other members of the training batch.
WEEK 2	The following topics were covered from this text book TAW ABAP Workbench Fundamentals 10_1 with theoretical knowledge and then practical work with solving exercises at the end of each unit using the ABAP Workbench to have a better understanding - • Unit 1: SAP Solutions • Unit 2: Navigation • Unit 3: The System Kernel • Unit 4: Communication and Integration Technologies • Unit 5: Sources of Information for Developers • Unit 6: ABAP Program Process • Unit 7: Introduction to ABAP Workbench • Unit 8: Basic ABAP Language Elements • Unit 9: Data Retrieval Developing summary report on my theoretical knowledge about SAP by the end of the week.
WEEK 3	The following topics were covered from this text book TAW ABAP Workbench Fundamentals 10_1 with theoretical knowledge and then practical work with solving exercises at the end of each unit using the ABAP Workbench to have a better understanding - • Unit 10: Subprograms in ABAP • Unit 11: Introduction to ABAP Events • Unit 12: Classic List Processing • Unit 13: Creating and Calling Function Groups and Function • Modules • Unit 14: Programs Calls and Data Storage Management

	The following topics were covered from this text book TAW ABAP Workbench Fundamentals 10_2 with theoretical knowledge and then practical work with solving exercises at the end of each unit using the ABAP Workbench to have a better understanding – • Unit 15: ABAP Dictionary • Unit 16: Tables in ABAP Dictionary • Unit 17: Performance When Accessing Tables • Unit 18: Input Checks • Unit 19: Dependencies with ABAP Dictionary Objects • Unit 20: Changing Tables
WEEK 4	The following topics were covered from this text book TAW ABAP Workbench Fundamentals 10_2 with theoretical knowledge and then practical work with solving exercises at the end of each unit using the ABAP Workbench to have a better understanding - • Unit 21: Views and Maintenance Dialogs • Unit 22: Search Help • Unit 23: ABAP Runtime • Unit 24: ABAP Types and Data Objects • Unit 25: Analysis Tools for Programs • Unit 26: ABAP Open SQL • Unit 27: Dynamic Programming Assignment of first module to be developed titled 'Dynamic Stock Status Report'. Creating logic, coding and debugging the program code finished by the end of this week.
WEEK 5	After assignment of first module, development continues. Debugging and commenting code. Preparing design report as deliverable to supervisor. Module demonstration to supervisor for approval. The following topics were covered from this text book TAW ABAP Workbench Fundamentals 12_1 with theoretical knowledge and then practical work with solving exercises at the end of each unit using the ABAP Workbench to have a better understanding – • Unit 28: Introduction to Screen Programming • Unit 29: The Program Interface

WEEK 6	The following topics were covered from this text book TAW ABAP Workbench Fundamentals 12_1 with theoretical knowledge and then practical work with solving exercises at the end of each unit using the ABAP Workbench to have a better understanding – • Unit 30: Output Elements • Unit 31: Input/Output Elements • Unit 32: Subscreens and Tabstrip Controls Elements • Unit 33: Table Controls Elements • Unit 34: Context Menus • Unit 35: Lists on Screens
WEEK 7	Assignment of second module to be developed titled 'Flight and Booking Information System'. Creating logic, coding and debugging the program code finished. Debugging and commenting the code finished by the end of this week.
WEEK 8	○ Preparing design report for the second module as deliverable to supervisor. ○ Preparing final report and demonstration for the working modules for the project manager. ○ Revision of comments made to code according to Wipro standards. ○ Final approval on successful completion of training and collecting certificate from project manager.

Appendix B – Program Code

Project 1 – Dynamic Stock Status Report

```
*&---------------------------------------------------------------------*
*& Report  ZSK_EXERCISE1
*& Dynamic Stock Status Report
*&---------------------------------------------------------------------*
*& Author: Shaeq Khan
*& Date: 2nd August 2011 to 10th August 2011
*&---------------------------------------------------------------------*

report  zsk_exercise1.

include zsk_exercise1_top.

include zsk_exercise1_subroutines.

" Include following standard tables in the module
tables: mara, marc, mard, mchb, mbew.

" Selection screen definitions
select-options: ma for mara-matnr obligatory,          " Material Number
                pl for marc-werks,                     " Plant
                sl for mard-lgort,                     " Storage Location
                ba for mchb-charg,                     " Batch
                ev for mbew-bwtar.                     " Valuation Type

**********************************************************************

" On request of F4 help by user, following functions are executed
" for select option fields 'low' and 'high'
at selection-screen on value-request for ma-low.
  perform sub_help_matnrl.

at selection-screen on value-request for ma-high.
  perform sub_help_matnrl.

at selection-screen on value-request for pl-low.
  perform sub_help_werks.

at selection-screen on value-request for pl-high.
  perform sub_help_werks.

at selection-screen on value-request for sl-low.
  perform sub_help_lgort.

at selection-screen on value-request for sl-high.
  perform sub_help_lgort.

at selection-screen on value-request for ba-low.
  perform sub_help_charg.
```

```
at selection-screen on value-request for ba-high.
  perform sub_help_charg.

at selection-screen on value-request for ev-low.
  perform sub_help_bwtar.

at selection-screen on value-request for ev-high.
  perform sub_help_bwtar.

*********************************************************************

" Event occuring after SELECTION-SCREEN has been processed.
start-of-selection.

  " If the 'Material Number' is entered by the user, the function 'ZSEL_MAT
NR'
  " is called to populate the internal table for the final list.
  if ma is not initial.
    call function 'ZSEL_MATNR'
      exporting
        p_ma      = ma[]            " Material Number
        p_pl      = pl[]            " Plant
        p_sl      = sl[]            " Storage Location
        p_ba      = ba[]            " Batch
        p_ev      = ev[]            " Valuation Type
      importing
        itab_list = itab_list.      " Internal table for the final list.


  " Else if 'Plant' is entered by the user, the function 'ZSEL_WERKS'
  " is called to populate the internal table for the final list.
  elseif pl is not initial.
    call function 'ZSEL_WERKS'
      exporting
        p_ma      = ma[]            " Material Number
        p_pl      = pl[]            " Plant
        p_sl      = sl[]            " Storage Location
        p_ba      = ba[]            " Batch
        p_ev      = ev[]            " Valuation Type
      importing
        itab_list = itab_list.      " Internal table for the final list.


  " Else if 'Storage Location' is entered by the user, the function 'ZSEL_L
GORT'
  " is called to populate the internal table for the final list.
  elseif sl is not initial.
    call function 'ZSEL_LGORT'
      exporting
        p_ma      = ma[]            " Material Number
        p_pl      = pl[]            " Plant
        p_sl      = sl[]            " Storage Location
```

```
        p_ba      = ba[]            " Batch
        p_ev      = ev[]            " Valuation Type
      importing
        itab_list = itab_list.      " Internal table for the final list.


  " Else if 'Batch' is entered by the user, the function 'ZSEL_CHARG'
  " is called to populate the internal table for the final list.
  elseif ba is not initial.
    call function 'ZSEL_CHARG'
      exporting
        p_ma      = ma[]            " Material Number
        p_pl      = pl[]            " Plant
        p_sl      = sl[]            " Storage Location
        p_ba      = ba[]            " Batch
        p_ev      = ev[]            " Valuation Type
      importing
        itab_list = itab_list.      " Internal table for the final list.


  " Else if 'Valuation Type' is entered by the user, the function 'ZSEL_BWT
AR'
  " is called to populate the internal table for the final list.
  elseif ev is not initial.
    call function 'ZSEL_BWTAR'
      exporting
        p_ma      = ma[]            " Material Number
        p_pl      = pl[]            " Plant
        p_sl      = sl[]            " Storage Location
        p_ba      = ba[]            " Batch
        p_ev      = ev[]            " Valuation Type
      importing
        itab_list = itab_list.      " Internal table for the final list.

  endif.

*********************************************************************

  " Header for the list.
  write: /000(010) 'MatCode',
         011(030) 'MatDesc',
         032(010) 'OldMatCode',
         043(005) 'UOM',
         049(015) 'Plant',
         065(010) 'MatType',
         076(015) 'MatGroup',
         092(015) 'ExtMatGrp',
         108(004) 'StorageLoc',
         113(010) 'ValType',
         124(010) 'Batch'.


  if itab_list is not initial.
```

```
    loop at itab_list into wa_list.

      write: /000(010) wa_list-mat hotspot,                " MatCode
             011(030) wa_list-mat_desc,                   " MatDesc
             032(010) wa_list-old_mat_code,               " OldMatCode
             043(005) wa_list-uom,                        " UOM
             049(015) wa_list-plant hotspot,              " Plant
             065(010) wa_list-mat_typ,                    " MatType
             076(015) wa_list-mat_grp,                    " MatGroup
             092(015) wa_list-ex_mat_grp,                 " ExtMatGrp
             108(004) wa_list-loc hotspot,                " StorageLoc
             113(010) wa_list-val_typ,                    " ValType
             124(010) wa_list-bat hotspot.                " Batch

      " Store the content of these variables and their line numbers
      hide: wa_list-mat, wa_list-plant, wa_list-loc, wa_list-bat.

    endloop.

  endif.

************************************************************************

at line-selection.

  " Record location of the cursor click
  get cursor field loc_cursor.

  if loc_cursor = 'WA_LIST-MAT'.
    set parameter id: 'MAT' field wa_list-mat,           " Material Number
                      'WRK' field wa_list-plant,         " Plant
                      'LAG' field wa_list-
loc,            " Storage Location
                      'CHA' field wa_list-bat.           " Batch

    call transaction 'MMBE' and skip first screen.
  endif.


************************************************************************

*&---------------------------------------------------------------------*
*&  Include           ZSK_EXERCISE1_TOP
*&
*&
*&  Global data declarations.
*&---------------------------------------------------------------------*

" Store location of field of the cursor click.
data  loc_cursor(20) type c .

" Structure for the final list to be generated
```

```
types: begin of struct_list,
        mat          type mara-matnr,        " Material Number
        mat_desc     type makt-maktx,        " Material Description
        old_mat_code type mara-bismt,        " Old Material Code
        uom          type mara-meins,        " Base Unit of Measurement
        plant        type marc-werks,        " Plant
        mat_typ      type mara-mtart,        " Material Type
        mat_grp      type mara-matkl,        " Material Group
        ex_mat_grp   type mara-extwg,        " External Material Group
        loc          type mard-lgort,        " Storage Location
        val_typ      type mbew-bwtar,        " Valuation Type
        bat          type mchb-charg,        " Batch
        end of struct_list.

data: itab_list type table of struct_list,   " Internal table
      wa_list   like line of itab_list.      " Work area

**********************************************************************

" Structure for the data from standard table MARA
types: begin of struct_mara,
        mat          type mara-matnr,        " Material Number
        old_mat_code type mara-bismt,        " Old Material Code
        uom          type mara-meins,        " Base Unit of Measurement
        mat_typ      type mara-mtart,        " Material Type
        mat_grp      type mara-matkl,        " Material Group
        ex_mat_grp   type mara-extwg,        " External Material Group
        end of struct_mara.

data: itab_mara type table of struct_mara,   " Internal table
      wa_mara   like line of itab_mara.      " Work area

**********************************************************************

" Structure for the data from the standard table MAKT
types: begin of struct_makt,
        mat          type makt-matnr,        " Material Number
        mat_desc     type makt-maktx,        " Material Description
       end of struct_makt.

data: itab_makt type table of struct_makt,   " Internal table
      wa_makt   like line of itab_makt.      " Work area

**********************************************************************

" Structure for the data from the standard table MARC
types: begin of struct_marc,
        mat          type marc-matnr,        " Material Number
        plant        type marc-werks,        " Plant
       end of struct_marc.

data: itab_marc type table of struct_marc,   " Internal table
      wa_marc   like line of itab_marc.      " Work area
```

```
*---------------------------------------------------------------------*

" Structure for the data from the standard table MARD
types: begin of struct_mard,
        mat           type mard-matnr,        " Material Number
        plant         type mard-werks,        " Plant
        loc           type mard-lgort,        " Storage Location
       end of struct_mard.

data: itab_mard type table of struct_mard,   " Internal table
      wa_mard like line of itab_mard.        " Work area

*---------------------------------------------------------------------*

" Structure for the data from the standard table MCHB
types: begin of struct_mchb,
        mat           type mchb-matnr,        " Material Number
        plant         type mchb-werks,        " Plant
        loc           type mchb-lgort,        " Storage Location
        bat           type mchb-charg,        " Batch
       end of struct_mchb.

data: itab_mchb type table of struct_mchb,   " Internal table
      wa_mchb like line of itab_mchb.        " Work area

*---------------------------------------------------------------------*

" Strucure for the data from the standard table MBEW
types: begin of struct_mbew,
        mat           type mbew-matnr,        " Material Number
        val_typ       type mbew-bwtar,        " Valuation Type
       end of struct_mbew.

data: itab_mbew type table of struct_mbew,   " Internal table
      wa_mbew like line of itab_mbew.        " Work area

*---------------------------------------------------------------------*

*&---------------------------------------------------------------------*
*&  Include           ZSK_EXERCISE1_SUBROUTINES
*&---------------------------------------------------------------------*


*&---------------------------------------------------------------------*
*&      Form  SUB_HELP_MATNRL
*&---------------------------------------------------------------------*
*       F4 Help for the field 'Material Number'
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
form sub_help_matnrl .
```

```
  " Structure to store all values of 'Material Numbers' from standard table
 MARA
  types: begin of struct_matnr,
          matnr type mara-matnr,               " Material Number
         end of struct_matnr.

  " Data Decaration
  data: itabm type table of struct_matnr.      " Internal Table

  select matnr                                 " Select 'Material Number'
    from mara                                  " from the standard table MAR
A
    into table itabm.                          " into the internal table

  call function 'F4IF_INT_TABLE_VALUE_REQUEST'
    exporting
     retfield               = 'matnr'          " Specify the internal table
field whose value will be returned to the screen field
     dynpprog               = sy-cprog         " Name of the program
     dynpnr                 = sy-dynnr         " Number of current screens
     dynprofield            = 'ma'             " Screen field on which the F
4 help is attached
     value_org              = 'S'              " Value return C: cell by cel
l, S: structured
    tables
     value_tab              = itabm            " Internal table with the val
ues
            .

endform.                    " SUB_HELP_MATNRL



*&---------------------------------------------------------------------*
*&      Form  SUB_HELP_WERKS
*&---------------------------------------------------------------------*
*       F4 Help for the field 'Plant'
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
form sub_help_werks .

  " Structure to store all values of 'Plant' from standard table MARC
  types: begin of struct_werks,
          werks type marc-werks,                " Plant
         end of struct_werks.

  " Data Decaration
  data itabw type table of struct_werks.        " Internal Table

  select werks                                  " Select 'Plant'
```

```
    from marc                                    " from the standard table M
ARC
    into table itabw.                            " into the internal table

  call function 'F4IF_INT_TABLE_VALUE_REQUEST'
    exporting
     retfield               = 'werks'            " Specify the internal tabl
e field whose value will be returned to the screen field
     dynpprog               = sy-cprog           " Name of the program
     dynpnr                 = sy-dynnr           " Number of current screens
     dynprofield            = 'pl'               " Screen field on which the
 F4 help is attached
     value_org              = 'S'                " Value return C: cell by c
ell, S: structured
    tables
     value_tab              = itabw              " Internal table with the v
alues
          .

endform.                    " SUB_HELP_WERKS



*&---------------------------------------------------------------------*
*&      Form  SUB_HELP_LGORT
*&---------------------------------------------------------------------*
*       F4 Help for the field 'Storage Location'
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
form sub_help_lgort .

  " Structure to store all values of 'Storage Location' from standard table
 MARD
  types: begin of struct_lgort,
          lgort type mard-lgort,               " Storage Location
         end of struct_lgort.

  " Data Decaration
  data itabl type table of struct_lgort.       " Internal Table

  select lgort                                 " Select 'Storage Location'
    from mard                                  " from the standard table M
ARD
    into table itabl.                          " into the internal table

  call function 'F4IF_INT_TABLE_VALUE_REQUEST'
    exporting
     retfield               = 'lgort'            " Specify the internal tabl
e field whose value will be returned to the screen field
     dynpprog               = sy-cprog           " Name of the program
     dynpnr                 = sy-dynnr           " Number of current screens
```

```
     dynprofield             = 'sl'               " Screen field on which the
 F4 help is attached
     value_org               = 'S'                " Value return C: cell by c
ell, S: structured
    tables
     value_tab               = itabl              " Internal table with the v
alues
            .

endform.                     " SUB_HELP_LGORT



*&---------------------------------------------------------------------*
*&      Form  SUB_HELP_CHARG
*&---------------------------------------------------------------------*
*       F4 Help for the field 'Batch'
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
form sub_help_charg .

  " Structure to store all values of 'Batch' from standard table MCHB
  types: begin of struct_charg,
          charg type mchb-charg,               " Batch
         end of struct_charg.

  " Data Decaration
  data itabc type table of struct_charg.       " Internal Table

  select charg                                 " Select 'Batch'
    from mchb                                  " from the standard table M
CHB
    into table itabc.                          " into the internal table

  call function 'F4IF_INT_TABLE_VALUE_REQUEST'
    exporting
     retfield                = 'charg'           " Specify the internal tabl
e field whose value will be returned to the screen field
     dynpprog                = sy-cprog          " Name of the program
     dynpnr                  = sy-dynnr          " Number of current screens
     dynprofield             = 'ba'              " Screen field on which the
 F4 help is attached
     value_org               = 'S'               " Value return C: cell by c
ell, S: structured
    tables
     value_tab               = itabc             " Internal table with the v
alues
            .

endform.                     " SUB_HELP_CHARG
```

```
*&---------------------------------------------------------------------*
*&      Form  SUB_HELP_BWTAR
*&---------------------------------------------------------------------*
*       F4 Help for the field 'Valuation Type'
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
form sub_help_bwtar .

  " Structure to store all values of 'Batch' from standard table MCHB
  types: begin of struct_bwtar,
          bwtar type mbew-bwtar,                " Valuation Type
         end of struct_bwtar.

  " Data Decaration
  data itabb type table of struct_bwtar.        " Internal Table

  select bwtar                                  " Select 'Batch'
    from mbew                                   " from the standard table M
BEW
    into table itabb.                           " into the internal table

  call function 'F4IF_INT_TABLE_VALUE_REQUEST'
    exporting
     retfield               = 'bwtar'           " Specify the internal tabl
e field whose value will be returned to the screen field
     dynpprog               = sy-cprog          " Name of the program
     dynpnr                 = sy-dynnr          " Number of current screens
     dynprofield            = 'ev'              " Screen field on which the
 F4 help is attached
     value_org              = 'S'               " Value return C: cell by c
ell, S: structured
    tables
     value_tab              = itabb             " Internal table with the v
alues
           .

endform.                    " SUB_HELP_BWTAR


function zsel_matnr.
*"----------------------------------------------------------------------
*"*"Local Interface:
*"  IMPORTING
*"     REFERENCE(P_MA) TYPE  ZSK_MATNR_TAB
*"     REFERENCE(P_PL) TYPE  ZSK_WERKS_TAB
*"     REFERENCE(P_SL) TYPE  ZSK_LGORTS_TAB
*"     REFERENCE(P_BA) TYPE  ZSK_CHARG_TAB
*"     REFERENCE(P_EV) TYPE  ZSK_BWTAR_TAB
*"  EXPORTING
```

```
*"     REFERENCE(ITAB_LIST) TYPE  ZSK_LIST_TAB
*"----------------------------------------------------------------------

 select matnr                          " Material Number
        bismt                          " Old Material Code
        meins                          " Base Unit of Measurement
        mtart                          " Material Type
        matkl                          " Material Group
        extwg                          " External Material Group
        from mara                      " From the standard table MARA
        into table itab_mara           " Into the internal table
        where matnr in p_ma.           " On condition that 'Material Number' i
s the same as the value entered.

 select matnr                          " Material Number
        maktx                          " Material Description
        from makt                      " From the standard table MAKT
        into table itab_makt           " Into the internal table
        where matnr in p_ma.           " On condition that 'Material Number' i
s the same as the value entered.

 select matnr                          " Material Number
        werks                          " Plant
        from marc                      " From the standard table MARC
        into table itab_marc           " Into the internal table
        where matnr in p_ma            " On condition that 'Material Number' i
s the same as the value entered.
        and   werks in p_pl.           " And 'Plant' is the same as the value
entered.

 select matnr                          " Material Number
        werks                          " Plant
        lgort                          " Storage Location
        from mard                      " From the standard table MARD
        into table itab_mard           " Into the internal table
        where matnr in p_ma            " On condition that 'Material Number' i
s the same as the value entered
        and   werks in p_pl            " And 'Plant' is the same as the value
entered
        and   lgort in p_sl.           " And 'Storage Location' is the same as
 the value entered.

 select matnr                          " Material Number
        werks                          " Plant
        lgort                          " Storage Location
        charg                          " Batch
        from mchb                      " From the standard table MCHB
        into table itab_mchb           " Into the internal table
        where matnr in p_ma            " On condition that 'Material Number' i
s the same as the value entered
        and   werks in p_pl            " And 'Plant' is the same as the value
entered
        and   lgort in p_sl            " And 'Storage Location' is the same as
```

```
 the value entered
        and   charg in p_ba.         " And 'Batch' is the same as the value
entered.

 select matnr                        " Material Number
        bwtar                        " Valuation Type
        from mbew                    " From the standard table MBEW
        into table itab_mbew         " Into the internal table
        where matnr in p_ma          " On condition that 'Material Number' i
s the same as the value entered
        and   bwtar in p_ev.         " And 'Valuation Type' is the same as t
he value entered.


loop at itab_mara into wa_mara.

  move-
corresponding wa_mara to wa_list.            " Move structure being looped
at into final structure
  wa_list-mat          = wa_mara-mat.
  wa_list-old_mat_code = wa_mara-old_mat_code.
  wa_list-uom          = wa_mara-uom.
  wa_list-mat_typ      = wa_mara-mat_typ.
  wa_list-mat_grp      = wa_mara-mat_grp.
  wa_list-ex_mat_grp   = wa_mara-ex_mat_grp.

  read table itab_makt with key mat = wa_list-
mat   " Read internal table with 'Material Number' as a key
                              into wa_makt.       " Into a work area.
  if sy-
subrc = 0.                                  " If the statement is successfu
lly executed
    wa_list-mat_desc = wa_makt-
mat_desc.            " copy desired fields into the final structure.
  endif.

  read table itab_marc with key mat = wa_list-
mat   " Read internal table with 'Material Number' as a key
                              into wa_marc.       " Into a work area.
  if sy-
subrc = 0.                                  " If the statement is successfu
lly executed
    wa_list-plant = wa_marc-
plant.                  " copy desired fields into the final structure.
  endif.

  read table itab_mard with key mat = wa_list-
mat       " Read internal table with 'Material Number' and
                              plant = wa_list-plant    " 'Plant' as a key
                              into wa_mard.           " Into a work area.
  if sy-
subrc = 0.                                      " If the statement is succe
ssfully executed
```

```
    wa_list-loc = wa_mard-
loc.                          " copy desired fields into the final structur
e.
  endif.

  read table itab_mchb with key mat = wa_list-
mat       " Read internal table with 'Material Number' and
                                plant = wa_list-plant    " 'Plant' and
                                loc = wa_list-
loc       " 'Storage Location' as a key
                                into wa_mchb.            " Into a work area.
  if sy-
subrc = 0.                                          " If the statement is succe
ssfully executed
    wa_list-bat = wa_mchb-
bat.                          " copy desired fields into the final structur
e.
  endif.

  read table itab_mbew with key mat = wa_list-
mat       " Read internal table with 'Material Number' and
                                into wa_mbew.            " Into a work area.
  if sy-
subrc = 0.                                          " If the statement is succe
ssfully executed
    wa_list-val_typ = wa_mbew-
val_typ.                  " copy desired fields into the final structure.
  endif.

  append wa_list to itab_list.                           " Append the struct
ure to the final internal table

  endloop.

endfunction.

function zsel_werks.
*"----------------------------------------------------------------------
*"*"Local Interface:
*"  IMPORTING
*"     REFERENCE(P_MA) TYPE  ZSK_MATNR_TAB
*"     REFERENCE(P_PL) TYPE  ZSK_WERKS_TAB
*"     REFERENCE(P_SL) TYPE  ZSK_LGORTS_TAB
*"     REFERENCE(P_BA) TYPE  ZSK_CHARG_TAB
*"     REFERENCE(P_EV) TYPE  ZSK_BWTAR_TAB
*"  EXPORTING
*"     REFERENCE(ITAB_LIST) TYPE  ZSK_LIST_TAB
*"----------------------------------------------------------------------

  select matnr                              " Material Number
         werks                              " Plant
         from marc                          " From the standard table MARC
         into table itab_marc               " Into the internal table
```

```
        where werks in p_pl                " On condition that 'Plant' is th
e same as the value entered
        and   matnr in p_ma.               " And 'Material Number' is the sa
me as the value entered.

  if itab_marc is not initial.

    select matnr                           " Material Number
           bismt                           " Old Material Code
           meins                           " Base Unit of Measurement
           mtart                           " Material Type
           matkl                           " Material Group
           extwg                           " External Material Group
           from mara                       " From the standard table MARA
           into table itab_mara            " Into the internal table
           for all entries in itab_marc
           where matnr = itab_marc-
mat.    " On condition that 'Material Number' is the same as the value enter
ed.

    select matnr                           " Material Number
           maktx                           " Material Description
           from makt                       " From the standard table MAKT
           into table itab_makt            " Into the internal table
           for all entries in itab_marc
           where matnr = itab_marc-
mat.    " On condition that 'Material Number' is the same as the value enter
ed.

    select matnr                           " Material Number
           werks                           " Plant
           lgort                           " Storage Location
           from mard                       " From the standard table MARD
           into table itab_mard            " Into the internal table
           for all entries in itab_marc
           where matnr = itab_marc-
mat     " On condition that 'Material Number' is the same as the value enter
ed
           and   werks in p_pl.            " And 'Plant' is the same as the
value entered.

    select matnr                           " Material Number
           werks                           " Plant
           lgort                           " Storage Location
           charg                           " Batch
           from mchb                       " From the standard table MCHB
           into table itab_mchb            " Into the internal table
           for all entries in itab_marc
           where matnr = itab_marc-
mat     " On condition that 'Material Number' is the same as the value enter
ed
           and   werks in p_pl             " And 'Plant' is the same as the
value entered
```

```
           and   lgort in p_sl            " And 'Storage Location' is the s
ame as the value entered
           and   charg in p_ba.           " And 'Batch' is the same as the
value entered.

    select matnr                          " Material Number
           bwtar                          " Valuation Type
           from mbew                      " From the standard table MBEW
           into table itab_mbew           " Into the internal table
           for all entries in itab_marc
           where matnr = itab_marc-
mat    " On condition that 'Material Number' is the same as the value enter
ed
           and   bwtar in p_ev.           " And 'Valuation type' is the sam
e as the value entered.

  endif.

  loop at itab_marc into wa_marc.

    wa_list-mat   = wa_marc-mat.
    wa_list-plant = wa_marc-plant.

    read table itab_mara with key mat = wa_marc-
mat       " Read internal table with 'Material Number' as a key
                                  into wa_mara.          " Into a work are
a.
    if sy-
subrc = 0.                                        " If the statement is succe
ssfully executed copy desired fields into the final structure.
      wa_list-old_mat_code = wa_mara-old_mat_code.
      wa_list-uom          = wa_mara-uom.
      wa_list-mat_typ      = wa_mara-mat_typ.
      wa_list-mat_grp      = wa_mara-mat_grp.
      wa_list-ex_mat_grp   = wa_mara-ex_mat_grp.
    endif.

    read table itab_makt with key mat = wa_list-
mat       " Read internal table with 'Material Number' as a key
                                  into wa_makt.          " Into a work are
a.
    if sy-
subrc = 0.                                        " If the statement is succe
ssfully executed copy desired fields into the final structure.
      wa_list-mat_desc = wa_makt-mat_desc.
    endif.

    read table itab_mard with key mat   = wa_list-
mat     " Read internal table with 'Material Number' and
                                  plant = wa_list-
plant   " And 'Plant' as key
                                  into wa_mard.          " Into a work are
a.
```

```
    if sy-
subrc = 0.                                          " If the statement is succe
ssfully executed copy desired fields into the final structure.
      wa_list-loc = wa_mard-loc.
    endif.

    read table itab_mchb with key mat = wa_list-
mat       " Read internal table with 'Material Number' and
                                    plant = wa_list-
plant    " And 'Plant' as key
                                    loc = wa_list-
loc       " And 'Storage Location' as key
                                    into wa_mchb.           " Into a work are
a.
    if sy-
subrc = 0.                                          " If the statement is succe
ssfully executed copy desired fields into the final structure.
      wa_list-bat = wa_mchb-bat.
    endif.

    read table itab_mbew with key mat = wa_list-
mat       " Read internal table with 'Material Number' as a key
                                    into wa_mbew.           " Into a work are
a.
    if sy-
subrc = 0.                                          " If the statement is succe
ssfully executed copy desired fields into the final structure.
      wa_list-val_typ = wa_mbew-val_typ.
    endif.

    append wa_list to itab_list.                        " Append the stru
cture to the final internal table

  endloop.

endfunction.

function zsel_lgort.
*"----------------------------------------------------------------------
*"*"Local Interface:
*"  IMPORTING
*"     REFERENCE(P_MA) TYPE  ZSK_MATNR_TAB
*"     REFERENCE(P_PL) TYPE  ZSK_WERKS_TAB
*"     REFERENCE(P_SL) TYPE  ZSK_LGORTS_TAB
*"     REFERENCE(P_BA) TYPE  ZSK_CHARG_TAB
*"     REFERENCE(P_EV) TYPE  ZSK_BWTAR_TAB
*"  EXPORTING
*"     REFERENCE(ITAB_LIST) TYPE  ZSK_LIST_TAB
*"----------------------------------------------------------------------

select  matnr                       " Material Number
        werks                       " Plant
        lgort                       " Storage Location
```

```
        from mard                  " From the standard table MARD
        into table itab_mard       " Into the internal table
        where lgort in p_sl.       " And 'Storage Location' is the same as t
he value entered.

if itab_mard is not initial.

  select  matnr                            " Material Number
          bismt                            " Old Material Code
          meins                            " Base Unit of Measurement
          mtart                            " Material Type
          matkl                            " Material Group
          extwg                            " External Material Group
          from mara                        " From the standard table MARA
          into table itab_mara             " Into the internal table
          for all entries in itab_mard
          where matnr = itab_mard-
mat.      " On condition that 'Material Number' is the same as the value en
tered.

  select  matnr                            " Material Number
          maktx                            " Material Description
          from makt                        " From the standard table MAKT
          into table itab_makt             " Into the internal table
          for all entries in itab_mard
          where matnr = itab_mard-
mat.      " On condition that 'Material Number' is the same as the value en
tered.

  select  matnr                            " Material Number
          werks                            " Plant
          from marc                        " From the standard table MARC
          into table itab_marc             " Into the internal table
          for all entries in itab_mard
          where matnr = itab_mard-
mat       " On condition that 'Material Number' is the same as the value en
tered.
          and   werks = itab_mard-
plant.    " And 'Plant' is the same as the value entered.

  select  matnr                            " Material Number
          werks                            " Plant
          lgort                            " Storage Location
          charg                            " Batch
          from mchb                        " From the standard table MCHB
          into table itab_mchb             " Into the internal table
          for all entries in itab_mard
          where matnr = itab_mard-
mat       " On condition that 'Material Number' is the same as the value en
tered
          and   werks = itab_mard-
plant     " And 'Plant' is the same as the value entered
          and   lgort in p_sl              " And 'Storage Location' is the
```

```
 same as the value entered
          and   charg in p_ba.                " And 'Batch' is the same as th
e value entered.

  select  matnr                               " Material Number
          bwtar                               " Valuation Type
          from mbew                           " From the standard table MBEW
          into table itab_mbew                " Into the internal table
          for all entries in itab_mard
          where matnr = itab_mard-
mat       " On condition that 'Material Number' is the same as the value en
tered
          and   bwtar in p_ev.                " And 'Valuation Type' is the s
ame as the value entered.

endif.

loop at itab_mard into wa_mard.

  wa_list-mat = wa_mard-mat.
  wa_list-loc = wa_mard-loc.

  read table itab_mara with key mat = wa_list-
mat         " Read internal table with 'Material Number' as a key
                                into wa_mara.             " Into a work are
a.
  if sy-
subrc = 0.                                    " If the statement is suc
cessfully executed
    wa_list-old_mat_code = wa_mara-
old_mat_code.           " copy desired fields into the final structure.
    wa_list-uom          = wa_mara-uom.
    wa_list-mat_typ      = wa_mara-mat_typ.
    wa_list-mat_grp      = wa_mara-mat_grp.
    wa_list-ex_mat_grp   = wa_mara-ex_mat_grp.
  endif.

  read table itab_makt with key mat = wa_list-
mat         " Read internal table with 'Material Number' as a key
                                into wa_makt.             " Into a work are
a.
  if sy-
subrc = 0.                                    " If the statement is suc
cessfully executed
    wa_list-mat_desc = wa_makt-
mat_desc.                  " copy desired fields into the final structure.
  endif.

  read table itab_marc with key mat = wa_list-
mat         " Read internal table with 'Material Number' as a key
                                into wa_marc.             " Into a work are
a.
  if sy-
```

```
subrc = 0.                                          " If the statement is suc
cessfully executed
    wa_list-plant = wa_marc-
plant.                          " copy desired fields into the final structur
e.
  endif.

  read table itab_mchb with key mat   = wa_list-
mat       " Read internal table with 'Material Number'
                                plant = wa_list-plant     " And 'Plant'
                                loc   = wa_list-
loc       " And 'Storage Location' as a key
                                into wa_mchb.             " Into a work are
a.
  if sy-
subrc = 0.                                          " If the statement is suc
cessfully executed
    wa_list-bat = wa_mchb-
bat.                            " copy desired fields into the final struct
ure.
  endif.

  read table itab_mbew with key mat = wa_list-
mat         " Read internal table with 'Material Number' as a key
                                into wa_mbew.             " Into a work are
a.
  if sy-
subrc = 0.                                          " If the statement is suc
cessfully executed
    wa_list-val_typ = wa_mbew-
val_typ.                     " copy desired fields into the final structure.
  endif.

append wa_list to itab_list.                              " Append the stru
cture to the final internal table

endloop.

endfunction.



function zsel_charg.
*"----------------------------------------------------------------------
*"*"Local Interface:
*"  IMPORTING
*"     REFERENCE(P_MA) TYPE  ZSK_MATNR_TAB
*"     REFERENCE(P_PL) TYPE  ZSK_WERKS_TAB
*"     REFERENCE(P_SL) TYPE  ZSK_LGORTS_TAB
*"     REFERENCE(P_BA) TYPE  ZSK_CHARG_TAB
*"     REFERENCE(P_EV) TYPE  ZSK_BWTAR_TAB
*"  EXPORTING
*"     REFERENCE(ITAB_LIST) TYPE  ZSK_LIST_TAB
*"----------------------------------------------------------------------
```

```
select  matnr                                  " Material Number
        werks                                  " Plant
        lgort                                  " Storage Location
        charg                                  " Batch
        from mchb                              " From the standard table MCHB
        into table itab_mchb                   " Into the internal table
        where charg in p_ba.                   " On condition 'Batch' is the s
ame as the value entered.

if itab_mchb is not initial.

  select  matnr                                " Material Number
          bismt                                " Old Material Code
          meins                                " Base Unit of Measurement
          mtart                                " Material Type
          matkl                                " Material Group
          extwg                                " External Material Group
          from mara                            " From the standard table MARA
          into table itab_mara                 " Into the internal table
          for all entries in itab_mchb
          where matnr = itab_mchb-
mat.      " On condition that 'Material Number' is the same as the value en
tered.

  select  matnr                                " Material Number
          maktx                                " Material Description
          from makt                            " From the standard table MAKT
          into table itab_makt                 " Into the internal table
          for all entries in itab_mchb
          where matnr = itab_mchb-
mat.      " On condition that 'Material Number' is the same as the value en
tered.

  select  matnr                                " Material Number
          werks                                " Plant
          from marc                            " From the standard table MARC
          into table itab_marc                 " Into the internal table
          for all entries in itab_mchb
          where matnr = itab_mchb-
mat.      " On condition that 'Material Number' is the same as the value en
tered.

  select  matnr                                " Material Number
          werks                                " Plant
          lgort                                " Storage Location
          from mard                            " From the standard table MARD
          into table itab_mard                 " Into the internal table
          for all entries in itab_mchb
          where matnr = itab_mchb-
mat       " On condition that 'Material Number' is the same as the value en
tered
          and   werks = itab_mchb-
```

```
plant.    " And 'Plant' is the same as the value entered

  select  matnr                                " Material Number
          bwtar                                " Valuation Type
          from mbew                            " From the standard table MBEW
          into table itab_mbew                 " Into the internal table
          for all entries in itab_mchb
          where matnr = itab_mchb-
mat.      " On condition that 'Material Number' is the same as the value en
tered

endif.

loop at itab_mchb into wa_mchb.

  wa_list-mat = wa_mchb-mat.
  wa_list-bat = wa_mchb-bat.

  read table itab_mara with key mat = wa_mchb-
mat         " Read internal table with 'Material Number' as a key
                                into wa_mara.             " Into a work are
a.
  if sy-
subrc = 0.                                          " If the statement is suc
cessfully executed
    wa_list-old_mat_code = wa_mara-
old_mat_code.          " copy desired fields into the final structure.
    wa_list-uom          = wa_mara-uom.
    wa_list-mat_typ      = wa_mara-mat_typ.
    wa_list-mat_grp      = wa_mara-mat_grp.
    wa_list-ex_mat_grp   = wa_mara-ex_mat_grp.
  endif.

  read table itab_makt with key mat = wa_mchb-
mat         " Read internal table with 'Material Number' as a key
                                into wa_makt.             " Into a work are
a.
  if sy-
subrc = 0.                                          " If the statement is suc
cessfully executed
    wa_list-mat_desc = wa_makt-
mat_desc.                  " copy desired fields into the final structure.
  endif.

  read table itab_marc with key mat = wa_mchb-
mat         " Read internal table with 'Material Number' as a key
                                into wa_marc.             " Into a work are
a.
  if sy-
subrc = 0.                                          " If the statement is suc
cessfully executed
    wa_list-plant = wa_marc-
plant.                        " copy desired fields into the final structur
```

```
e.
  endif.

  read table itab_mard with key mat    = wa_mchb-
mat       " Read internal table with 'Material Number' and
                                plant = wa_mchb-
plant     " And 'Plant' as a key
                                into wa_mard.             " Into a work are
a.
  if sy-
subrc = 0.                                          " If the statement is suc
cessfully executed
    wa_list-loc = wa_mard-
loc.                            " copy desired fields into the final struct
ure.
  endif.

  read table itab_mbew with key mat = wa_mchb-
mat         " Read internal table with 'Material Number' as a key
                                into wa_mbew.             " Into a work are
a.
  if sy-
subrc = 0.                                          " If the statement is suc
cessfully executed
    wa_list-val_typ = wa_mbew-
val_typ.                    " copy desired fields into the final structure.
  endif.

  append wa_list to itab_list.                              " Append the stru
cture to the final internal table

endloop.

endfunction.


function zsel_bwtar.
*"----------------------------------------------------------------------
*"*"Local Interface:
*"  IMPORTING
*"     REFERENCE(P_MA) TYPE  ZSK_MATNR_TAB
*"     REFERENCE(P_PL) TYPE  ZSK_WERKS_TAB
*"     REFERENCE(P_SL) TYPE  ZSK_LGORTS_TAB
*"     REFERENCE(P_BA) TYPE  ZSK_CHARG_TAB
*"     REFERENCE(P_EV) TYPE  ZSK_BWTAR_TAB
*"  EXPORTING
*"     REFERENCE(ITAB_LIST) TYPE  ZSK_LIST_TAB
*"----------------------------------------------------------------------

select  matnr                           " Material Number
        bwtar                           " Valuation Type
        from mbew                       " From the standard table MBEW
        into table itab_mbew            " Into the internal table
```

```
        where bwtar in p_ev.          " On condition that 'Valuation Type'
is the same as the value entered

if itab_mbew is not initial.

  select  matnr                               " Material Number
          bismt                               " Old Material Code
          meins                               " Base Unit of Measurement
          mtart                               " Material Type
          matkl                               " Material Group
          extwg                               " External Material Group
          from mara                           " From the standard table MARA
          into table itab_mara                " Into the internal table
          for all entries in itab_mbew
          where matnr = itab_mbew-
mat.      " On condition that 'Material Number' is the same as the value en
tered.

  select  matnr                               " Material Number
          maktx                               " Material Description
          from makt                           " From the standard table MAKT
          into table itab_makt                " Into the internal table
          for all entries in itab_mbew
          where matnr = itab_mbew-
mat.      " On condition that 'Material Number' is the same as the value en
tered.

  select  matnr                               " Material Number
          werks                               " Plant
          from marc                           " From the standard table MARC
          into table itab_marc                " Into the internal table
          for all entries in itab_mbew
          where matnr = itab_mbew-
mat.      " On condition that 'Material Number' is the same as the value en
tered.

  select  matnr                               " Material Number
          werks                               " Plant
          lgort                               " Storage Location
          from mard                           " From the standard table MARD
          into table itab_mard                " Into the internal table
          for all entries in itab_mbew
          where matnr = itab_mbew-
mat.      " On condition that 'Material Number' is the same as the value en
tered

  select  matnr                               " Material Number
          werks                               " Plant
          lgort                               " Storage Location
          charg                               " Batch
          from mchb                           " From the standard table MCHB
          into table itab_mchb                " Into the internal table
          for all entries in itab_mbew
```

```
          where matnr = itab_mbew-
mat.      " On condition that 'Material Number' is the same as the value en
tered

endif.

loop at itab_mbew into wa_mbew.

  wa_list-mat     = wa_mbew-mat.
  wa_list-val_typ = wa_mbew-val_typ.

  read table itab_mara with key mat = wa_list-
mat       " Read internal table with 'Material Number' as a key
                                into wa_mara.           " Into a work area.
  if sy-
subrc = 0.                                          " If the statement is succe
ssfully executed
    wa_list-old_mat_code = wa_mara-
old_mat_code.         " copy desired fields into the final structure.
    wa_list-uom          = wa_mara-uom.
    wa_list-mat_typ      = wa_mara-mat_typ.
    wa_list-mat_grp      = wa_mara-mat_grp.
    wa_list-ex_mat_grp   = wa_mara-ex_mat_grp.
  endif.

  read table itab_makt with key mat = wa_list-
mat       " Read internal table with 'Material Number' as a key
                                into wa_makt.           " Into a work area.
  if sy-
subrc = 0.                                          " If the statement is succe
ssfully executed
    wa_list-mat_desc = wa_makt-
mat_desc.               " copy desired fields into the final structure.
  endif.

  read table itab_marc with key mat = wa_list-
mat       " Read internal table with 'Material Number' as a key
                                into wa_marc.           " Into a work area.
  if sy-
subrc = 0.                                          " If the statement is succe
ssfully executed
    wa_list-plant = wa_marc-
plant.                      " copy desired fields into the final structure.
  endif.

  read table itab_mard with key mat   = wa_list-
mat     " Read internal table with 'Material Number'
                                plant = wa_list-
plant   " And 'Plant' as a key.
                                into wa_mard.           " Into a work area.
  if sy-
subrc = 0.                                          " If the statement is succe
ssfully executed
```

```
    wa_list-loc = wa_mard-
loc.                          " copy desired fields into the final structur
e.
  endif.

  read table itab_mchb with key mat    = wa_list-
mat     " Read internal table with 'Material Number'
                                plant = wa_list-plant    " And 'Plant'
                                loc   = wa_list-
loc     " And 'Storage Location' as a key.
                                into wa_mchb.            " Into a work area.
  if sy-
subrc = 0.                                        " If the statement is succe
ssfully executed
    wa_list-bat = wa_mchb-
bat.                          " copy desired fields into the final structur
e.
  endif.

  append wa_list to itab_list.                           " Append the struct
ure to the final internal table

endloop.

endfunction.
```

Project 2 – Flight and Booking Information

```
*&---------------------------------------------------------------------*
*& Module Pool       ZSK_EXERCISE2
*& Flight and Booking Information
*&---------------------------------------------------------------------*
*& Author: Shaeq Khan
*& Date: 20th August to 23rd August 2011
*&---------------------------------------------------------------------*

include zsk_exercise2_top                       .  " global Data

include zsk_exercise2_o01                       .  " PBO-Modules

include zsk_exercise2_i01                       .  " PAI-Modules

* INCLUDE ZSK_EXERCISE2_F01                      .  " FORM-Routines



*&---------------------------------------------------------------------*
*& Include ZSK_EXERCISE2_TOP                                 Module Pool
   ZSK_EXERCISE2
*&
*&---------------------------------------------------------------------*

program  zsk_exercise2 message-id zsk_mes.

" Include the follwing tables in the program.
tables: zsk_sflight, sbook.

" Structure to store flight details
types: begin of stru_flight,
        carrid     type sflight-carrid,         " Carrier ID
        connid     type sflight-connid,         " Connection ID
        fldate     type sflight-fldate,         " Flight Date
        currency   type sflight-currency,       " Currency
        planetype  type sflight-planetype,      " Plane Type
        seatsmax   type sflight-seatsmax,       " Maximum Seats
        seatsocc   type sflight-seatsocc,       " Occupied Seats
        sel        type c,                      " Selection marker. 'X' if
row is selected or ''
       end of stru_flight.

" Structure to store booking details
types: begin of stru_book,
        carrid     type sbook-carrid,           " Carrier ID
        connid     type sbook-connid,           " Connection ID
        fldate     type sbook-fldate,           " Flight Date
        bookid     type sbook-bookid,           " Booking ID
        customid   type sbook-customid,         " Customer ID
        custtype   type sbook-custtype,         " Customer Type
        class      type sbook-class,            " Ticket Class
        order_date type sbook-order_date,       " Order Date
```

```
       end of stru_book.

" Table control declaration for screen 100
controls: flight type tableview using screen 100.

" Data declaration
data: ok_code type sy-ucomm,
      save_ok type sy-ucomm,

      it_flight     type table of stru_flight,   " Internal table for struct
ure flight
      wa_flight     type stru_flight,            " Work area for structure f
light
      wa_flight_del type stru_flight,            " Another Work area for str
ucture flight

      it_book type table of stru_book,           " Internal table for struct
ure booking
      wa_book like line of it_book,              " Work area like a line of
the internal table

      carr_id type s_carr_id,                    " Parameter to record enter
ed detail from screen
      flag    type c value 'N',                  " Variable to check if the
table control should be enabled or disabled
      flagdel type c value 'N'.                  " Variable to record if the
 user has requested for deleting a row


*&---------------------------------------------------------------------*
*&  Include           ZSK_EXERCISE2_O01
*&---------------------------------------------------------------------*

*&---------------------------------------------------------------------*
*&      Module  DISPLAY_FLIGHT  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
module display_flight output.

  if flag = 'N'.       " Flag 'N' indicates that all screen elements should
be disabled
    loop at screen.
      if screen-group1 = 'GRP' or screen-name = 'WA_FLIGHT-CARRID'.
        screen-input = 0.      " Disabling all screen elements
        modify screen.
      endif.
      "  'GRP' is a group that contains the following screen variables
      "  Connection ID, Flight Date, Currency, Plane Type, Maximum Seats, O
ccupied Seats
    endloop.
  endif.
```

```
  if it_flight is not initial and flag = 'Y'.
    loop at screen.
      if screen-group1 = 'GRP'.
        screen-input = 1.      " Enabling all screen elements in 'GRP'
        modify screen.
      endif.
    endloop.
  endif.

endmodule.                 " DISPLAY_FLIGHT  OUTPUT


*&---------------------------------------------------------------------*
*&      Module  STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
*       text
*---------------------------------------------------------------------*
module status_0100 output.

  set pf-status 'STATUS100'.
  set titlebar 'FLIGHT'.

  " Initializing the table control with the exact number of lines as in the
 internal table
  describe table it_flight lines flight-lines.

endmodule.                 " STATUS_0100  OUTPUT


*&---------------------------------------------------------------------*
*&      Module  STATUS_0101  OUTPUT
*&---------------------------------------------------------------------*
*       text
*---------------------------------------------------------------------*
module status_0101 output.

  set pf-status 'STATUS101'.
  set titlebar 'STATUS101'.

endmodule.                 " STATUS_0101  OUTPUT


*&---------------------------------------------------------------------*
*&      Module  WRITE_LIST  OUTPUT
*&---------------------------------------------------------------------*
*       To display a list of all bookings for the selected flight
*---------------------------------------------------------------------*
module write_list output.

  loop at it_flight into wa_flight.
    if wa_flight-sel = 'X'.

      select carrid                                          " Carrier ID
```

```
              connid                                                  " Connection
ID
              fldate                                                  " Flight Date
              bookid                                                  " Booking ID
              customid                                                " Customer ID
              custtype                                                " Customer Ty
pe
              class                                                   " Ticket Clas
s
              order_date                                              " Order Date
              from sbook                                              " From the st
andard table SBOOK
              into corresponding fields of table it_book              " Into an int
ernal table
              where carrid = wa_flight-
carrid                    " On condition that the 'Carrier ID' is the same as
 the one for the selected row
              and   connid = wa_flight-
connid.                   " And the 'Connection ID' is the same as the one fo
r the selected row
*             AND   fldate = wa_flight-fldate.
    endif.
  endloop.

  " Header for the list.
  write: / 'CARRID',
           ' CONNID',
           '      FLDATE',
           '        BOOKID',
           '       CUSTID',
           '   CUSTYP',
           ' CLASS',
           '     ORDATE'.
          uline.

  " Looping into the internal table and displaying it.
  loop at it_book into wa_book.
    write: / wa_book-carrid, '    ',
             wa_book-connid, '    ',
             wa_book-fldate, '    ',
             wa_book-bookid, '    ',
             wa_book-customid, '     ',
             wa_book-custtype, '     ',
             wa_book-class,  '    ',
             wa_book-order_date.
  endloop.

  " Command to display a list on a screen in a module pool
  leave to list-processing and return to screen 0.
  suppress dialog.

endmodule.                    " WRITE_LIST  OUTPUT
```

```
*&---------------------------------------------------------------------*
*&  Include           ZSK_EXERCISE2_I01
*&---------------------------------------------------------------------*


*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
*       On user input
*---------------------------------------------------------------------*
module user_command_0100 input.

  save_ok = ok_code.
  clear ok_code.

  case save_ok.

    when 'BACK'.
      leave to screen '0'.

    when 'SEARCH'.
      " Re-initializing the flag variables.
      flag = 'N'.
      flagdel = 'N'.

      if carr_id is not initial.
        select carrid                       " Carrier ID
               connid                       " Connection ID
               fldate                       " Flight Date
               currency                     " Currency
               planetype                    " Plane Type
               seatsmax                     " Maximum Seats
               seatsocc                     " Occupied Seats
               from zsk_sflight             " From the table ZSK_SFLIGHT
               into table it_flight         " Into an internal table
               where carrid = carr_id.      " For the condition that the 'C
arrier ID' is the same as the one entered by the user
      endif.

    when 'SAVE'.

      if flagdel = 'Y'.     " If the user has clicked on delete and then as
ked to save then use 'DELETE' command
        delete zsk_sflight from wa_flight_del.
      else.                 " Else use the 'Modify' command to make changes
 to the database
        modify zsk_sflight from table it_flight.
      endif.

      message s000(zsk_mes) .
```

```
    when 'UPDATE'.
      " Change flag variable when user clicks on 'Update' button
      flag = 'Y'.


    when 'INSERT'.
      " Change flag variable when user clicks on 'Update' button
      flag = 'Y'.

      loop at it_flight into wa_flight.
        if wa_flight-
sel = 'X'.                      " If the row is selected by the user
          insert initial line into it_flight.       " Insert a new line abo
ve the selected line
        endif.
      endloop.


    when 'DELETE'.
      " Read the inernal table with the specified key into the work area
      read table it_flight with key sel = 'X' into wa_flight_del.
      " Delete the row which is selected by the user.
      delete it_flight where sel = 'X'.
      " Change the flag variable to effective save the changes made if the
user clicks 'Save'.
      flagdel = 'Y'.


    when 'VIEW'.
      call screen 101.

    when 'EXIT'.
      leave program.

    when 'CANCEL'.
      leave to screen '0'.

  endcase.

  clear save_ok.

endmodule.                 " USER_COMMAND_0100  INPUT


*&---------------------------------------------------------------------*
*&      Module  READ_DATA  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
module read_data.

  " The following statement changes any change(s) made to the internal tabl
```

```
e to the table control
  modify it_flight from wa_flight index flight-current_line.

endmodule.                    " READ_DATA  INPUT


*&---------------------------------------------------------------------*
*&      Module  EXIT  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
module exit input.

endmodule.                    " EXIT  INPUT


*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0101  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
module user_command_0101 input.

  case ok_code.

    when 'EXIT'.
      leave program.

    when 'CANCEL'.
      leave to screen '0'.

    when 'BACK'.
      leave to screen '100'.

  endcase.

endmodule.                    " USER_COMMAND_0101  INPUT


*&---------------------------------------------------------------------*
*&      Screen 0100
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
process before output.
  module status_0100.

  loop at it_flight into wa_flight with control flight.
    module display_flight.
  endloop.

*
process after input.
```

```
  module exit at exit-command.

  loop at it_flight.
    module read_data.
  endloop.

  module user_command_0100.


*&---------------------------------------------------------------------*
*&      Screen 0101
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
process before output.
  module status_0101.

  module write_list.
*
process after input.
  module user_command_0101.
```

Appendix C – Contact Information Form

Mobile Number

0091 999 524 3551 [Kochi, India]

00966 506 580 388 [Dhahran, KSA]

Email

s200792450@kfupm.edu.sa

shaeq.khan@wipro.com

shaeqkhan@gmail.com

Company/Institute

Wipro, KODC,

Info Park Special Economic Zone,

Kusumagiri PO, Kakkanad,

Kochi – 682 030, India.

Direct Supervisor:

Mr. Praveen Sharma, PMP, CISA

Project Manager, Delivery Manager

Email: praveen.sharma@wipro.com

Phone no: 0091 989 531 5140

Project Title

SAP ABAP Training and Project work.

Appendix D – Progress Reports

KING FAHD UNIV. OF PETROLEUM AND MINERALS
DEANSHIP OF STUDENT AFFAIRS
SUMMER TRAINING DEPARTMENT

جامعة الملك فهد للبترول و المعادن
عمادة شؤون الطلاب
إدارة التدريب الصيفي

Progress Report No. (1) تقرير مرحلي رقم

Student Name: SHAEQ PERVEZ KHAN	**KFUPM ID:** 200792450
Major: SOFTWARE ENGINEERING	**Date:** 26/07/2011

Brief description of the activities, assignments, projects, and training the student was involved in:

Studied the following concepts theoretically and practically by solving exercises under guidance and supervision [Source – ABAP Workbench Fundamentals, Participant Handbook]

- SAP Solutions
- The System Kernel
- ABAP Program Process
- ABAP Workbench
- Basic ABAP Language Elements
- Data Retrieval
- Subprograms in ABAP
- ABAP Events
- Classic List Processing
- Function Groups and Function Modules
- Data Storage Management
- ABAP Dictionary
- Input Checks
- Dependencies with ABAP Dictionary Objects
- Views and Maintenance Dialogs
- ABAP Runtime
- ABAP Types and Data Objects
- Analysis Tool for Programs
- ABAP OpenSQL
- Dynamic Programming

Supervisor Name: Praveen Sharma	**Signature:**
Position: Project Manager / Delivery Head	**Date:** 26/07/2011
Company/Organization: WIPRO LIMITED (Please affix company stamp)	**Phone:** **Fax:** 00919895315140 **E-Mail:** praveen.sharma@wipro.com

Important Instructions	تعليمات هامة
1. This form is to be filled in English by the student and approved by his supervisor at work. 2. This form is to be filled for three periods: after 3, 6, and 8 weeks. 3. The original approved three progress reports must be attached to the final report to be submitted to the academic department. 4. In case you need to type or extend this form, the supervisor should sign any additional pages.	١. يعبأ الطالب هذا النموذج بالإنجليزية ويصادق عليه المشرف في العمل. ٢. يعبأ هذا النموذج لثلاث فترات كالتالي: بعد ٣، ٦، ٨ أسابيع. ٣. يجب إرفاق النسخ الأصلية المصدقة لهذه التقارير الثلاثة مع كامل التقرير النهائي المقدم للقسم الأكاديمي. ٤. إذا أردت الزيادة أو الطباعة فيجب مصادقة المشرف على الملحقات.

KING FAHD UNIV. OF PETROLEUM AND MINERALS
DEANSHIP OF STUDENT AFFAIRS
SUMMER TRAINING DEPARTMENT

جامعة الملك فهد للبترول و المعادن
عمادة شؤون الطلاب
إدارة التدريب الصيفي

Progress Report No. (2) تقرير مرحلي رقم

Student Name: SHAEQ PERVEZ KHAN	**KFUPM ID:** 200792450
Major: SOFTWARE ENGINEERING	**Date:** 11/08/2011

Brief description of the activities, assignmemts, projects, and training the student was involved in:

Studied the following concepts theoretically and practically by solving exercises under guidance and supervision [Source – ABAP Workbench Fundamentals, Participant Handbook]

- Screen Programming
- Input/Output elements
- Subscreens

Completed the development of a report module with specified requirements which covered the following topics previously studied –

Data retrieval, subroutines, events, list processing, function modules and search helps

An outline of the module's requirements is attached.

Supervisor Name: Praveen Alex Luke	**Signature:**
Position: Project Manager	**Date:** 11/08/2011
Company/Organization: WIPRO Ltd. (Please affix company stamp)	**Phone:** +919895115941 **Fax:** **E-Mail:** praveen.luke@wipro.com

Important Instructions	تعليمات هامة
1. This form is to be filled in English by the student and approved by his supervisor at work.	١. يعبأ الطالب هذا النموذج بالإنجليزية ويصادق عليه المشرف في العمل.
2. This form is to be filled for three periods: after 3, 6, and 8 weeks.	٢. يعبأ هذا النموذج لثلاث فترات كالتالي: بعد ٣، ٦، ٨ أسابيع.
3. The original approved three progress reports must be attached to the final report to be submitted to the academic department.	٣. يجب إرفاق النسخ الأصلية المصدقة لهذه التقارير الثلاثة مع كامل التقرير النهائي المقدم للقسم الأكاديمي.
4. In case you need to type or extend this form, the supervisor should sign any additional pages.	٤. إذا أردت الزيادة أو الطباعة فيجب مصادقة المشرف على الملحقات.

Dynamic Stock Status Report

Selection Screen				Standard Table	Field
Material Code		To		MARA	MATNR
Plant		To		MARC	WERKS
Storage Location		To		MARD	LGORT
Batch		To		MCHB	CHARG
Valuation Type		To		MBEW	BWTAR

- User should be allowed to enter data in any of the fields
- Pass the combination of MATNR + WERKS + LGORT to get the CHARG
- Output should be according to the field specified by the user
- Make 'Material Code' as mandatory
- If we are passing more than one field i.e. Plant, Batch and Material Code, records satisfying all details should come in the list.

Output list should contain the following fields

1. Material
2. Material Description
3. Old Material Code
4. UOM
5. Plant
6. Material Type
7. Material Group
8. External Material Group
9. Storage Location
10. Valuation Type
11. Batch

Make 'Material Code' as a hotspot, clicking on it should take you to a transaction 'MMBE' (Stock Overview) bypassing selection screen for corresponding values of Plant, Batch, Storage Location and Material in that row.

Appendix E – Sample of Deliverables to the Company

DELIVERABLE 1 – Design report on module 1 'Dynamic Stock Status Report'

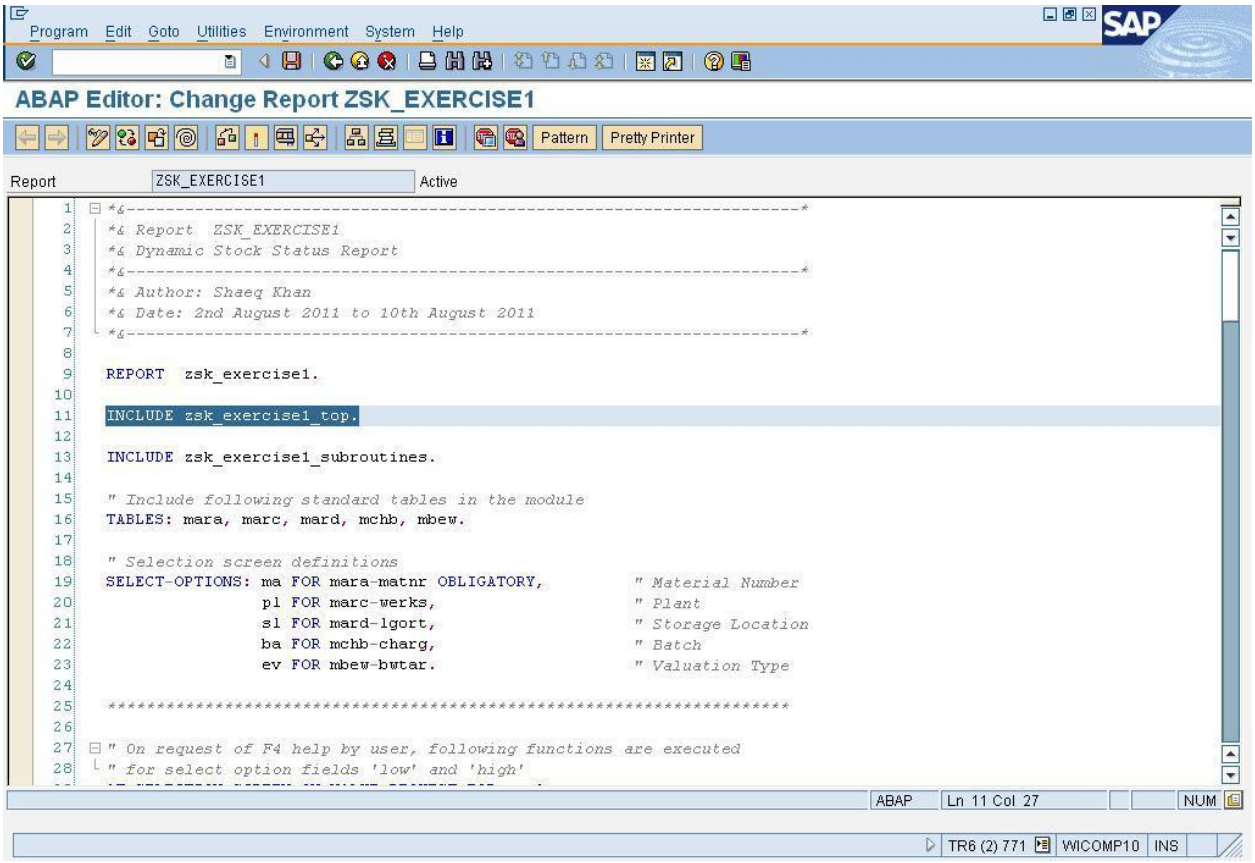

Figure 3 – Initial Screen on opening program

A basic structure of a report contains data declarations, database access and displaying the data required. In case where the module to be developed is big, developers use the concept of modularizations where similar functionalities are grouped together and then included in the main program. Here we are required to access six tables and store them in internal tables. Once stored, we retrieve the attributes that are required to display as a list output. One such attribute is marked in a way that if a user clicks on it, they are taken to another transaction triggered by the click event.

Let us go step by step into the code and understand how the entire module works.

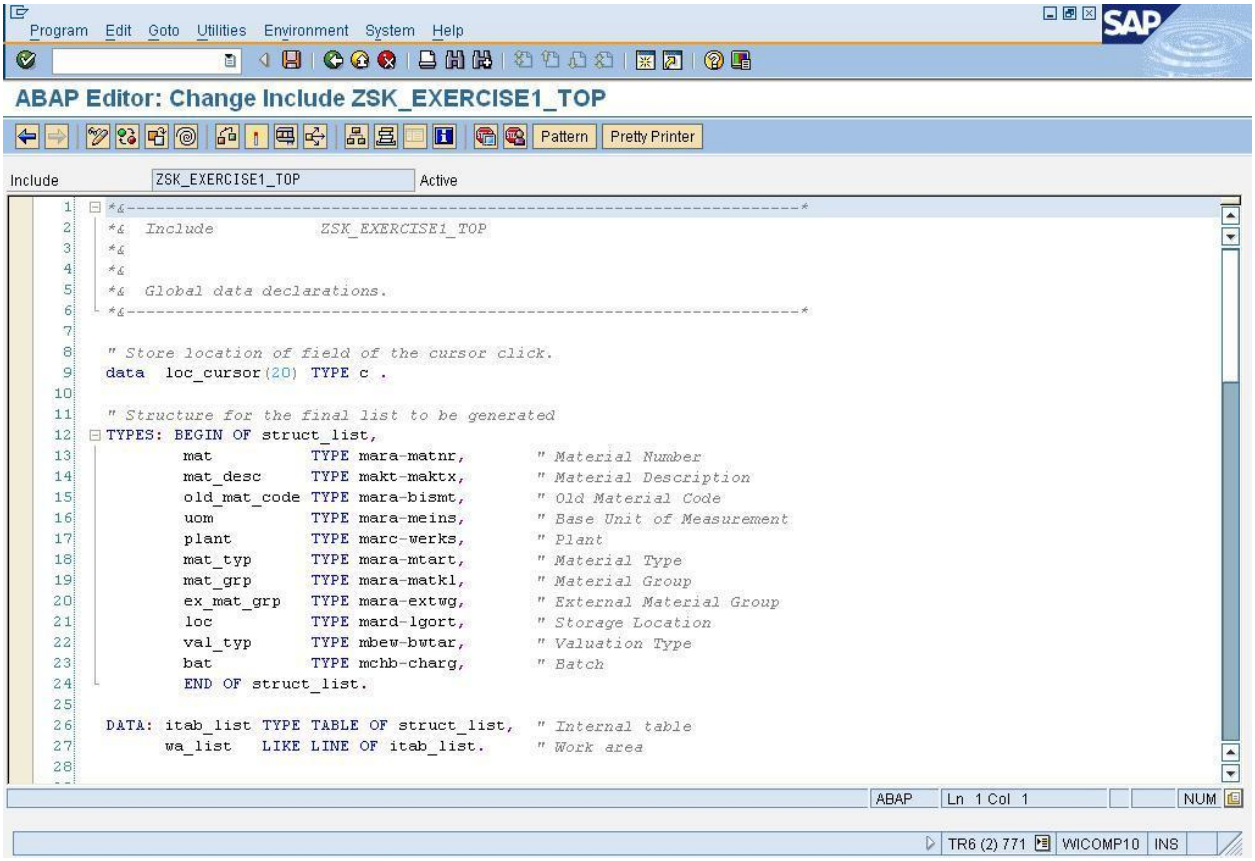

Figure 4 – Top include in the module

The TOP INCLUDE is a naming convention for the Include Type that contains all the global data declarations for the module. We define structures and internal tables to store and traverse through data that is read from tables stored in the repository.

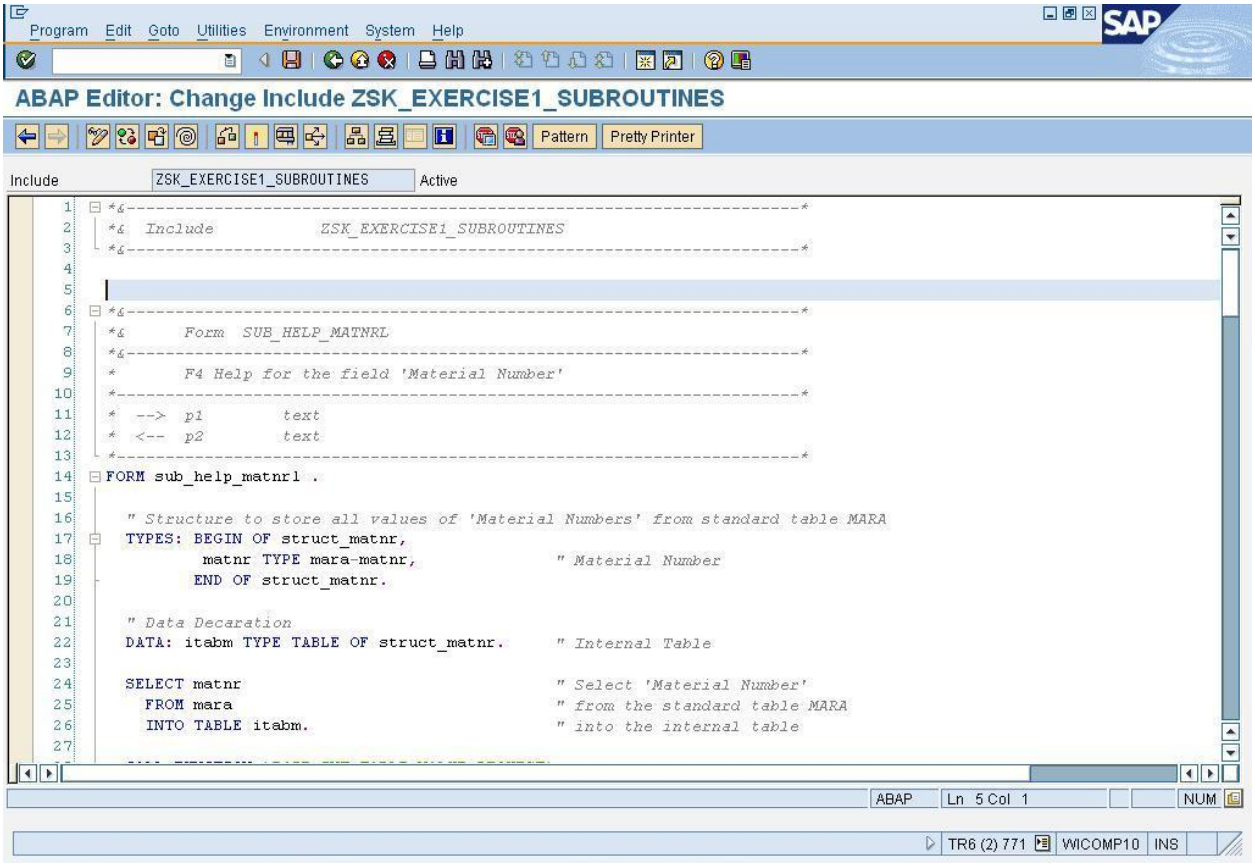

Figure 5 – Subroutines in the module

Since we employ the creation and use of several subroutines for this module, we created an Include Type that stores all the subroutines. A subroutine is a piece of code that performs certain functionality as required by the coder and then returns the flow to the main program.

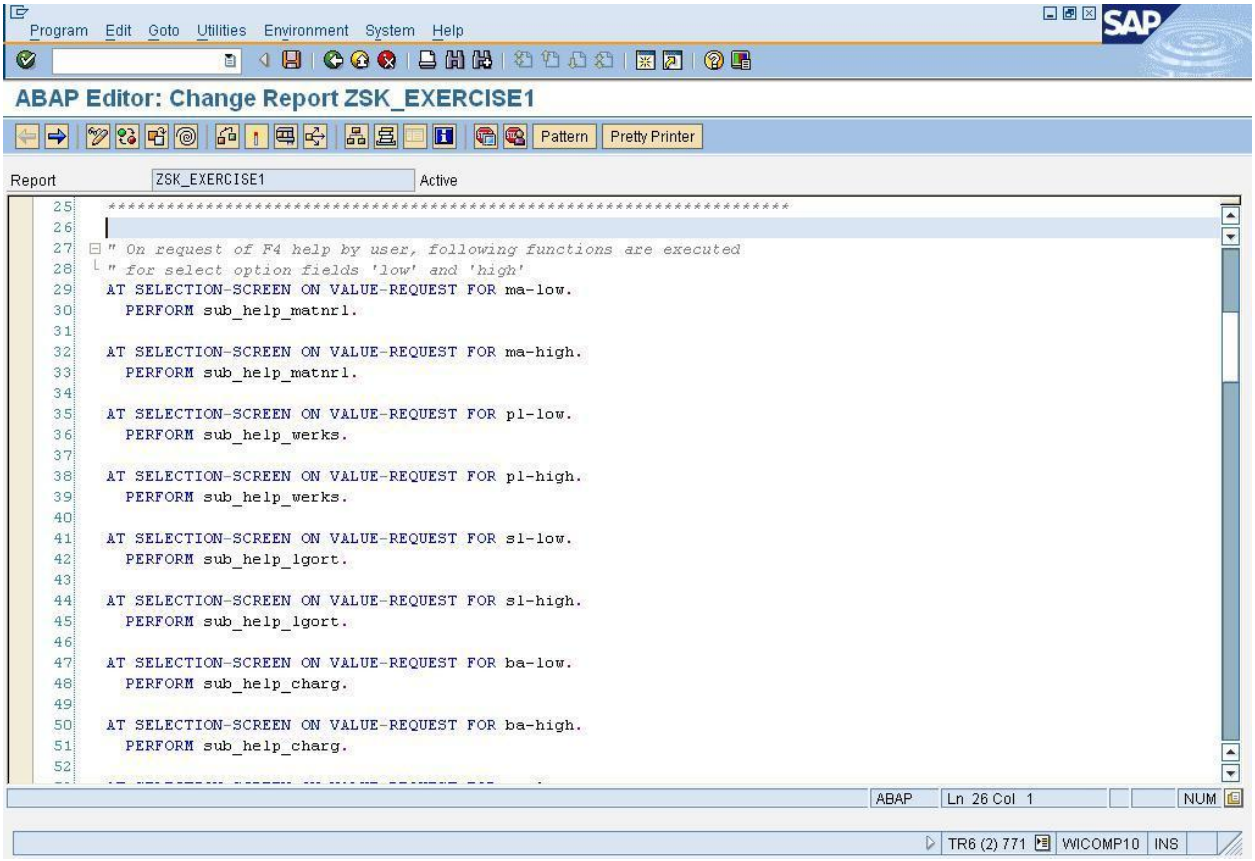

Figure 6 – Select Options to record user entry

Select-Options field is declared which enable user to enter data onto the screen. During this period if the user requires assistance with entering values into a field, they can press F4 to view possible entries for the field. This data is pulled in from the database that contains data for these fields. This functionality is achieved by writing a subroutine with a function call for a built in `'F4IF_INT_TABLE_VALUE_REQUEST'` function in SAP. The coder only has to specify the actual parameters and the help options are returned for the field for which the F4 help was initiated for. Let us have a look at the subroutine with the function call.

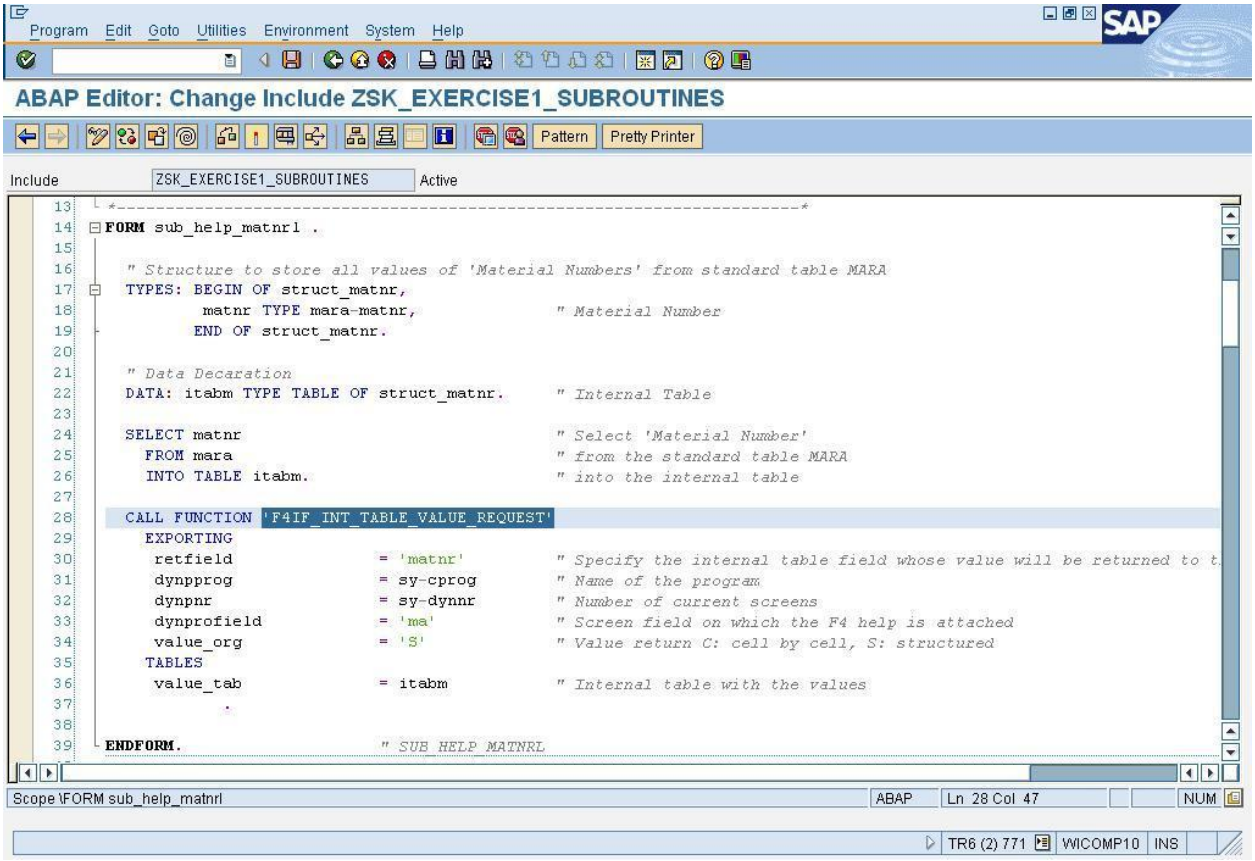

Figure 7 – 'F4IF_INT_TABLE_VALUE_REQUEST' function call

The shown subroutine saves all values of the field 'Material Code' [in the above shown case] into an internal table and then displays that table when the user requests for F4 help. The table handles events and the value clicked on by the user then appears on the field as the user's input. The table disappears if the user hits F4 again in case they don't require input help.

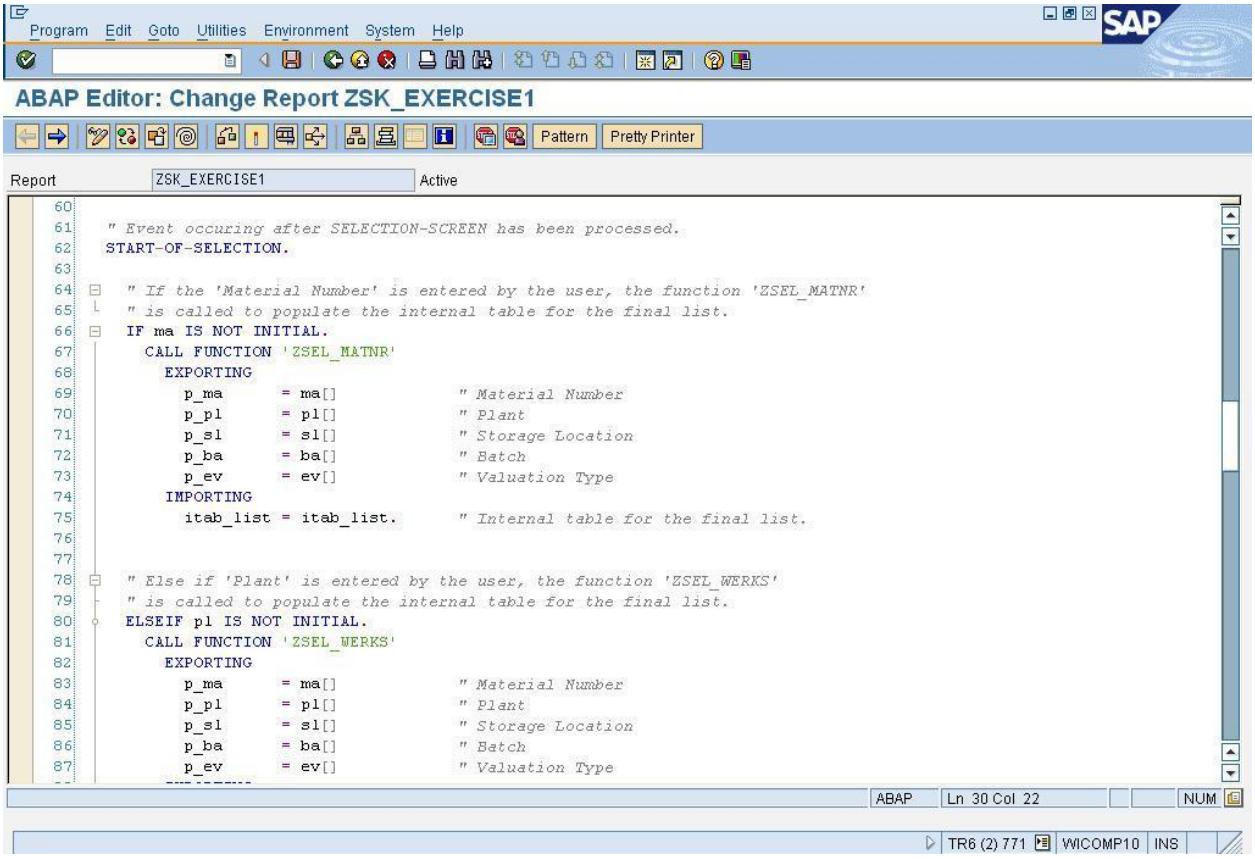

Figure 8 – Start of Selection

Upon the start of selection [after user has entered a value and executed the program again] we populate the final list to be displayed according to the fields that have been entered by the user. For that we use an If loop to check which fields are not still on their initial state. For the fields with value we execute a user declared function that retrieves value accordingly.

For example, if the user only enters 'Material Code' from the screen shot you can see that the function '`ZSEL_MATNR`' will be executed. Let us study the implemented function module in further detail.

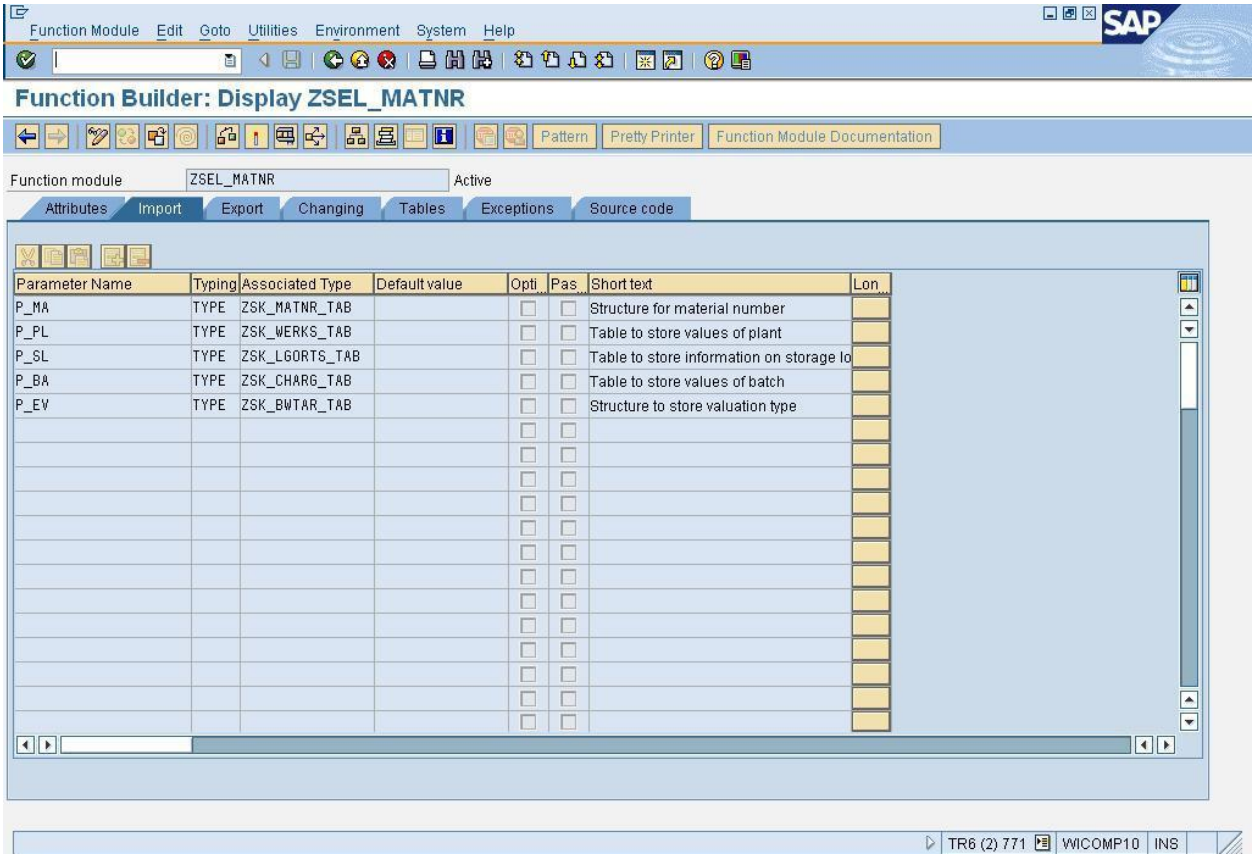

Figure 9 – Import/Export parameters for the function

In the Import tab we declare the parameters that we want to pass into the function.

In the Export tab we declare the parameters that will be passed back to the main program.

In the Source code we write in the logic of the function using the parameters declared in the Import and Export tabs.

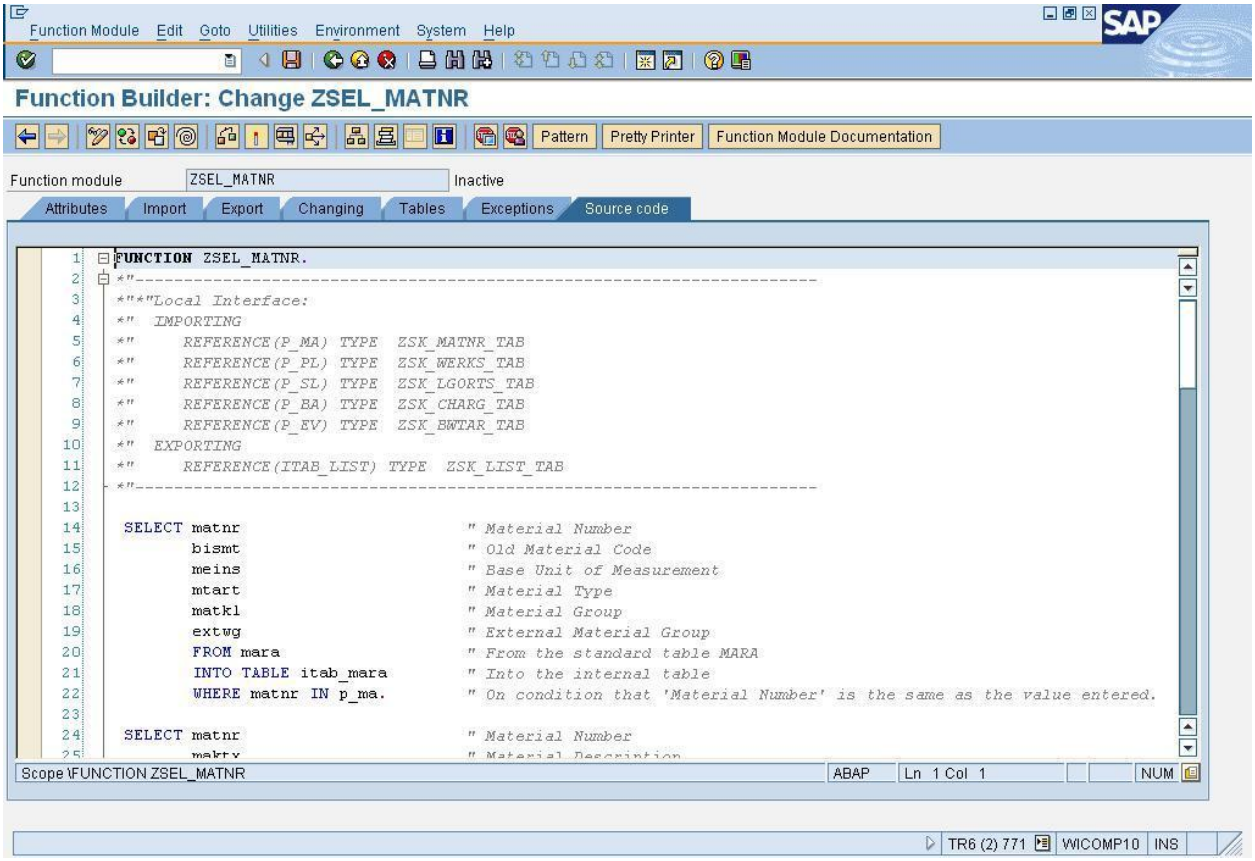

Figure 10 – Function source code containing the logic of the function

For this module the queries for each function call was written according to what data is available and what data is to be searched for. All the data retrieved was stored in an internal table that was passed back to the main program.

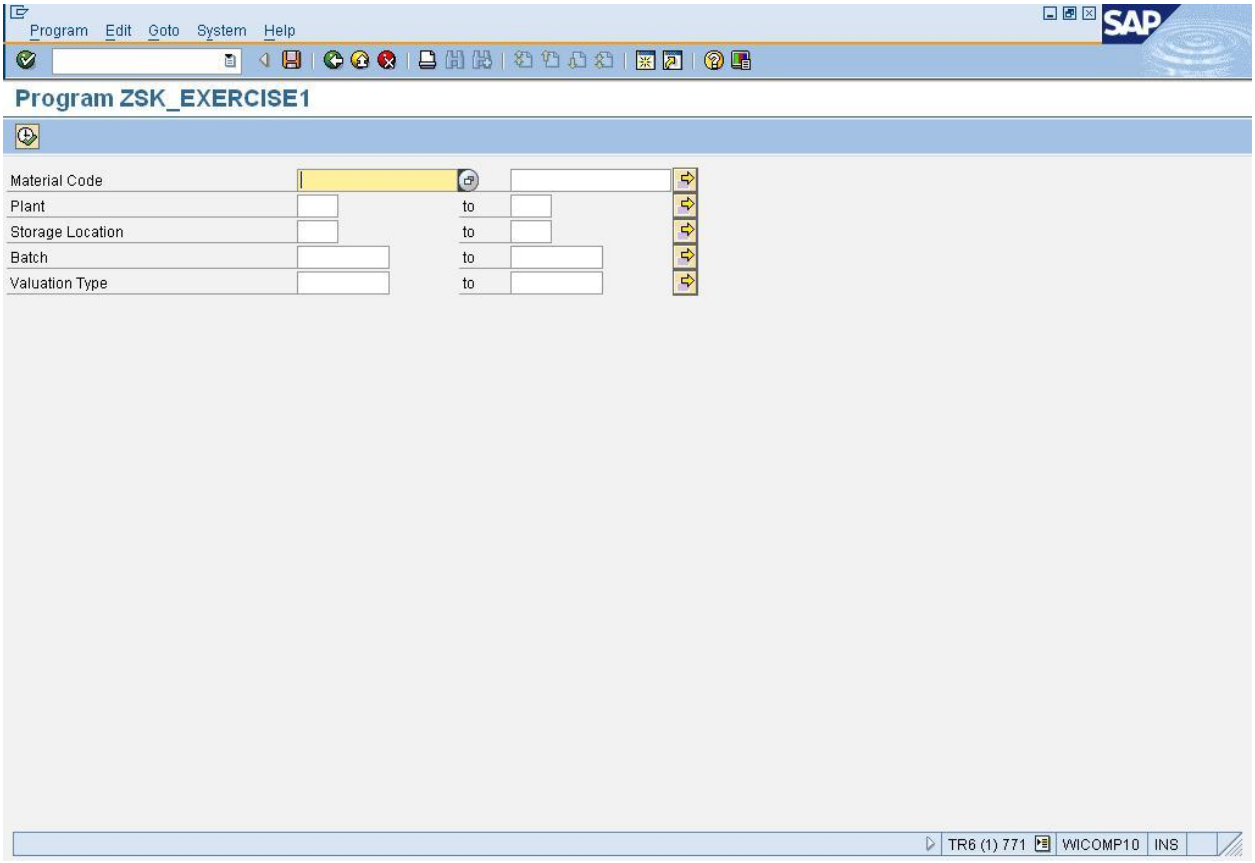

Figure 11 – Output Screen

Here user can enter values they require to check data for. In case the user needs to find a value, they can put the cursor on that particular field and press F4 to get input help as shown in the next screen shot.

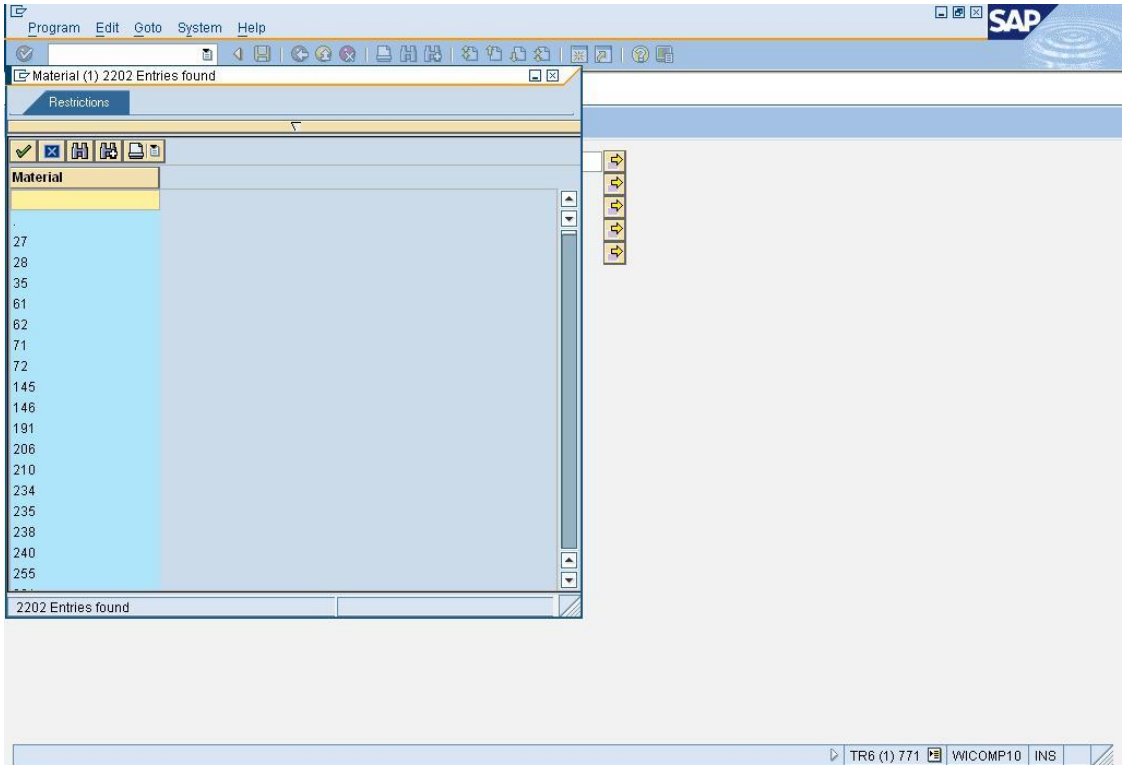

Figure 12 – Input Help for 'Material Code'

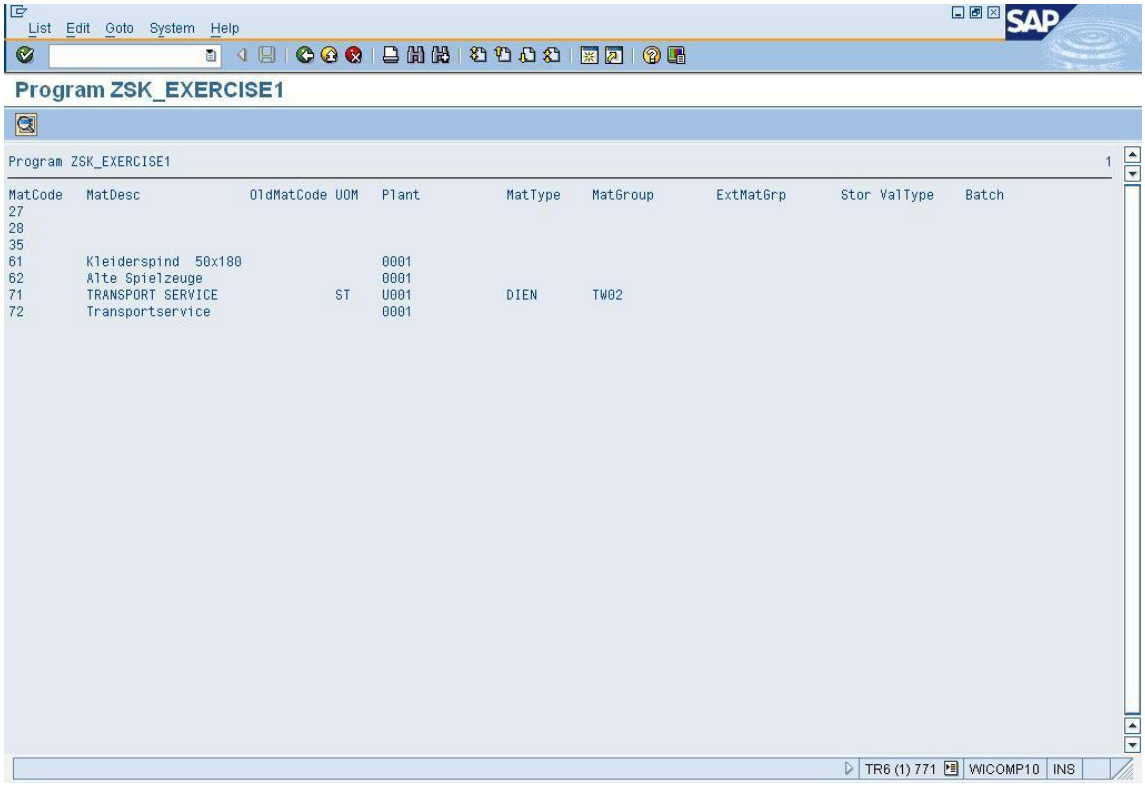

Figure 13 – The list containing data is displayed

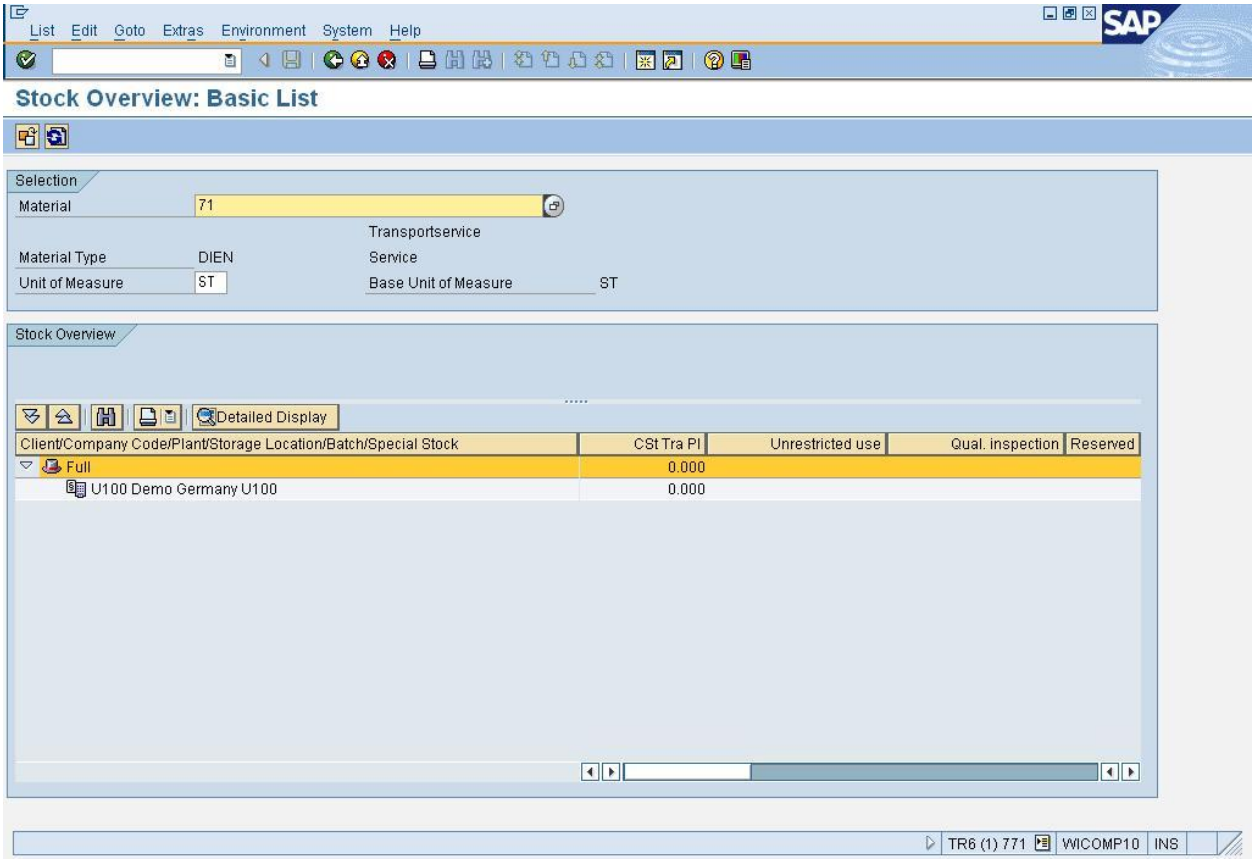

Figure 14 – Transaction MMBE called upon clicking on 'Material Code'

When the user clicks the hotspot area once [in this case the 'Material Code'], they are redirected to the MMBE transaction with copy of parameters that were entered on the initial input screen. These input fields are passed as parameters to the input fields of this transaction and hence available on reaching this screen.

DELIVERABLE 2 – Design report on module 2 'Flight and Booking Information System'

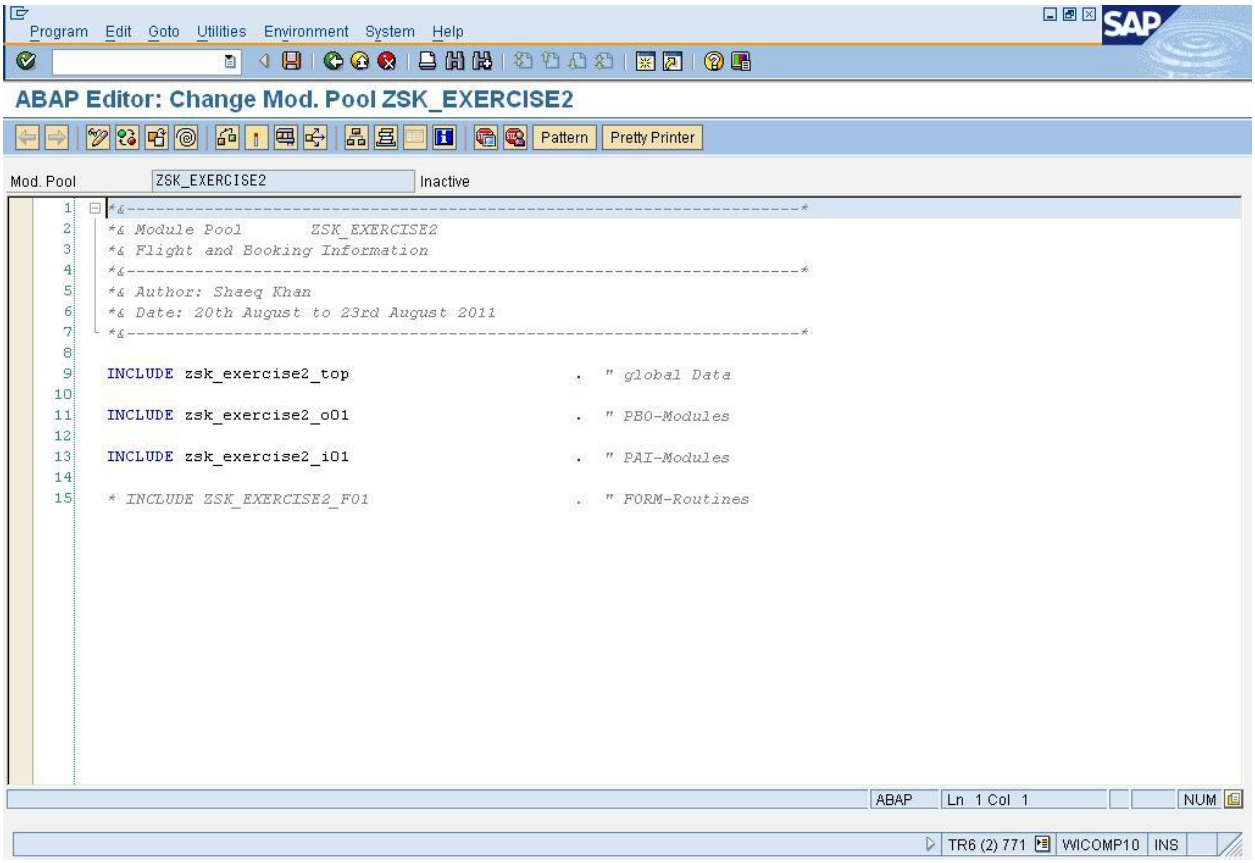

Figure 15 – Initial screen on opening program

A basic structure of a module pool contains data declarations, database access and defining the screen flow logic.

Let us go step by step into the code and understand how the entire module works.

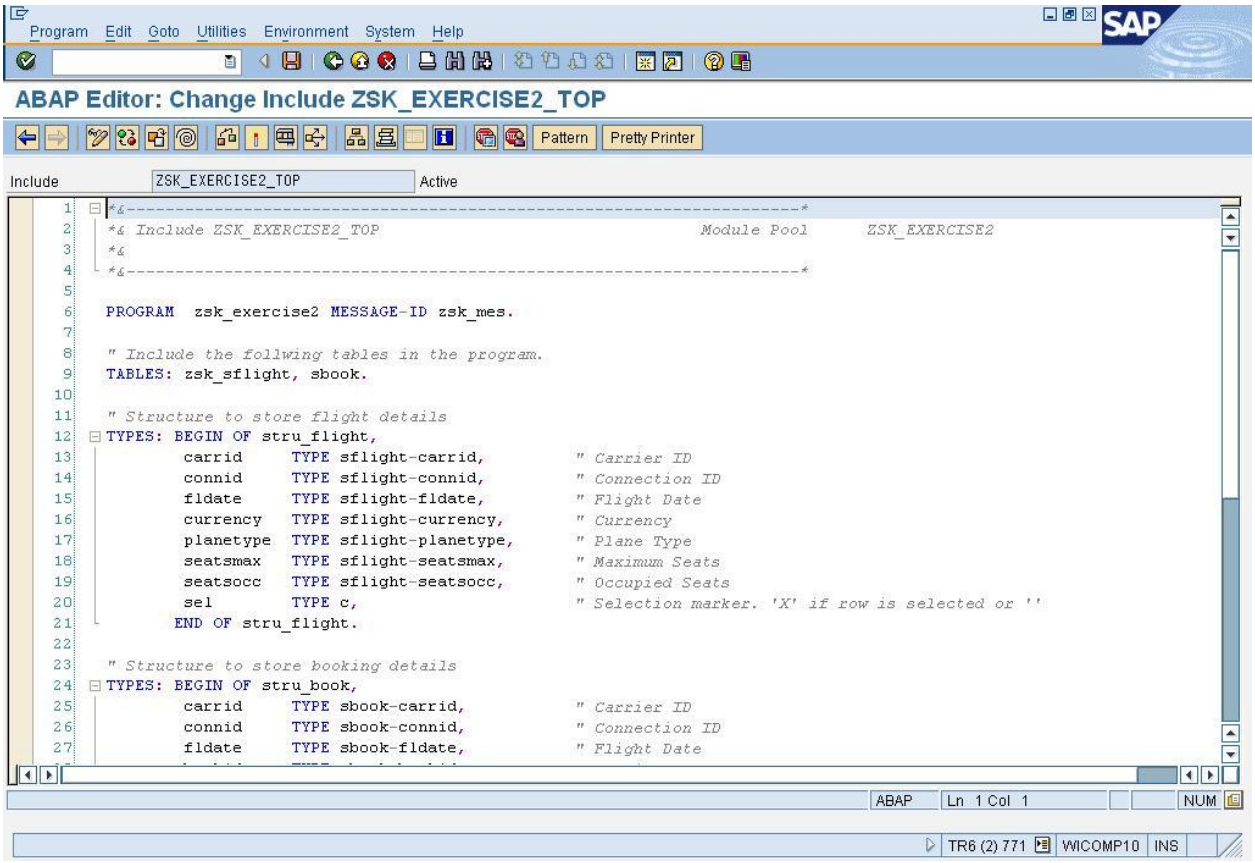

Figure 16 – The TOP INCLUDE in the program

The TOP INCLUDE is a naming convention for the Include Type that contains all the global data declarations for the module. We define structures and internal tables to store and traverse through data that is read from tables stored in the repository.

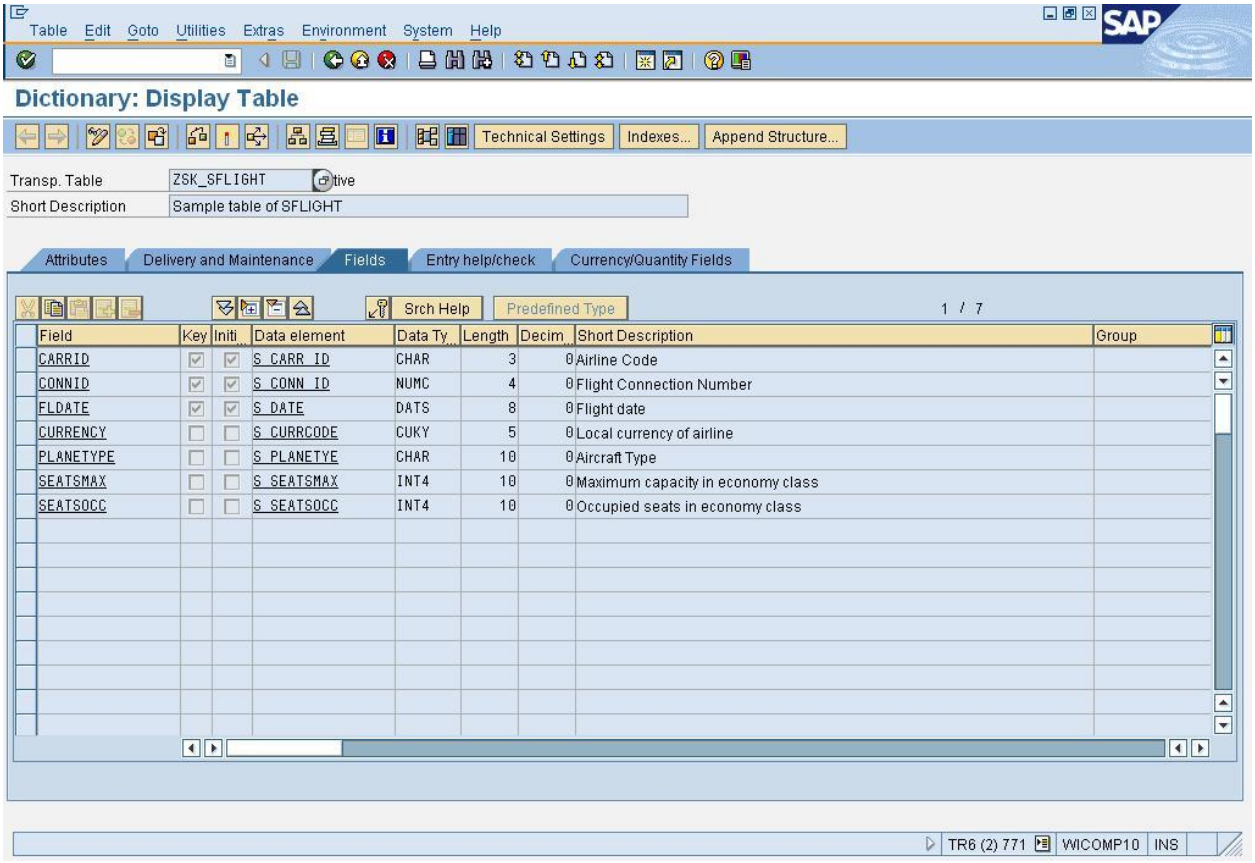

Figure 17 – Table for storing flight information

Created a Z table with selected attributes from the standard table SFLIGHT to store and update data. Created the table control on the screen using this Z table.

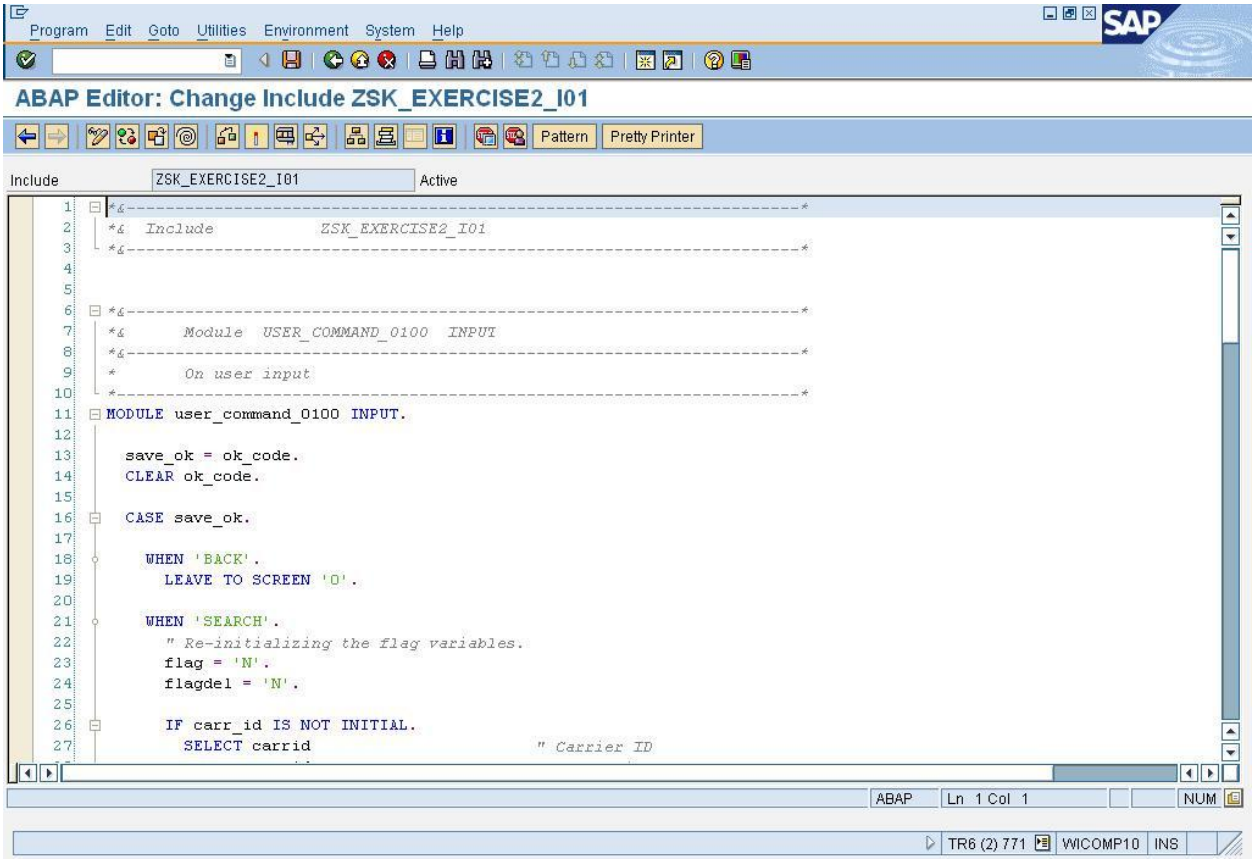

Figure 18 – The Process After Input [PAI] module

The PAI – Process After Input and the PBO – Process Before Output are dialog modules that form the screen flow logic together.

The PAI It is responsible for handling user inputs on the screen. The PBO updates the variables, internal tables and database before updating the screen.

The flow of information from the screen back to the screen is as follows –

Changes made on screen by user detected -> PAI module -> dialog processor -> PBO module -> update screen

The PAI module has a variable that records user inputs and perform operations accordingly. This is where we call other modules or forms which contain the core logic.

The PBO module contains the code to update all screen elements before they are displayed back to the user with updates made.

The dialog modules [PAI and PBO] are called on every user input.

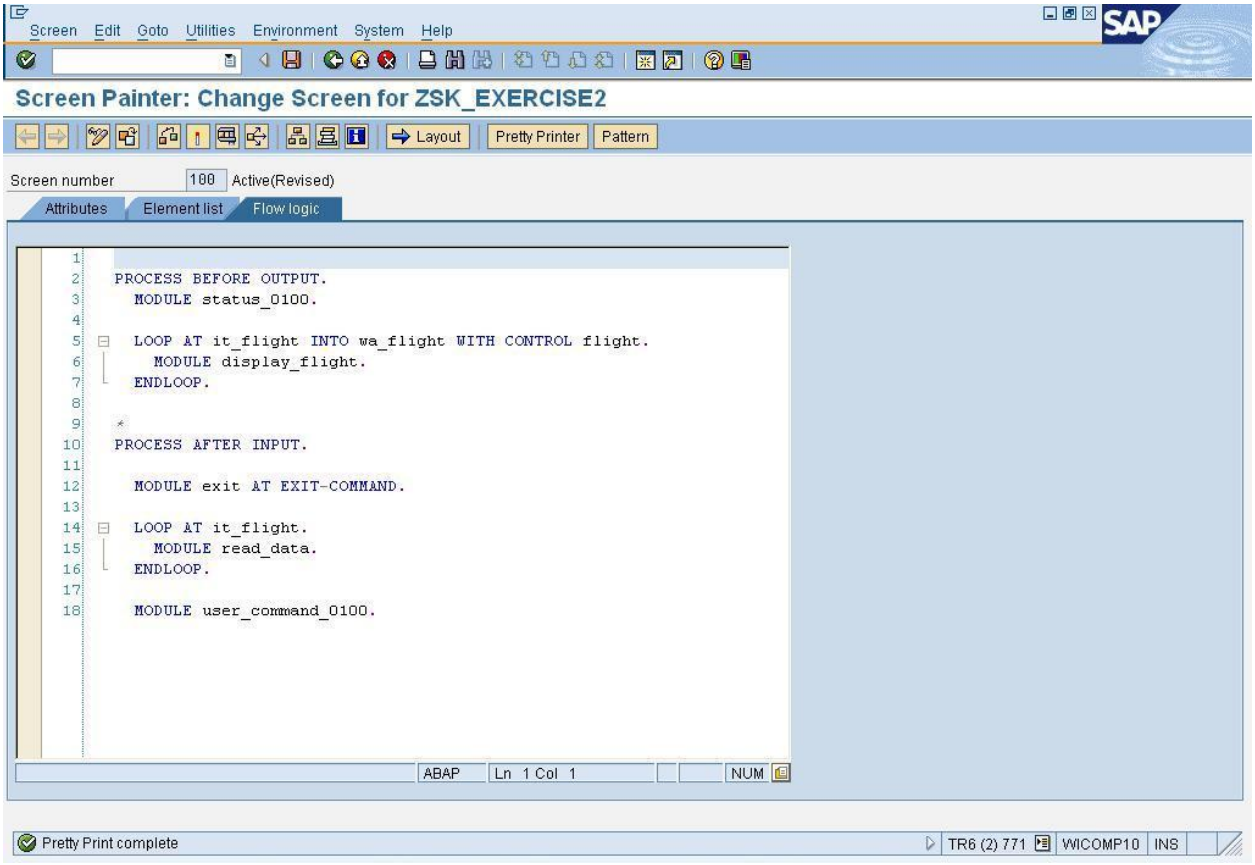

Figure 19 – Flow logic for screen 100

Each screen has its own flow logic. In this case screen 100 has a table control so we have loop over the table control in both PBO and PAI.

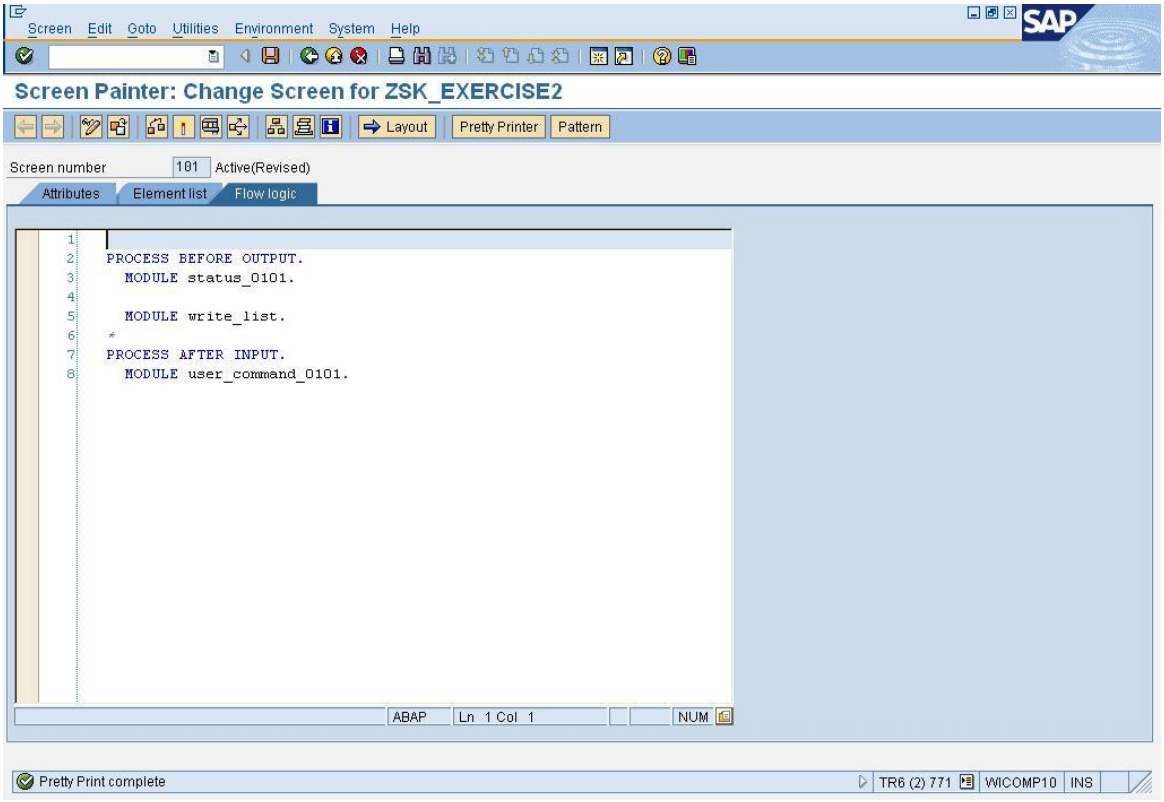

Figure 20 – Flow logic for screen 101

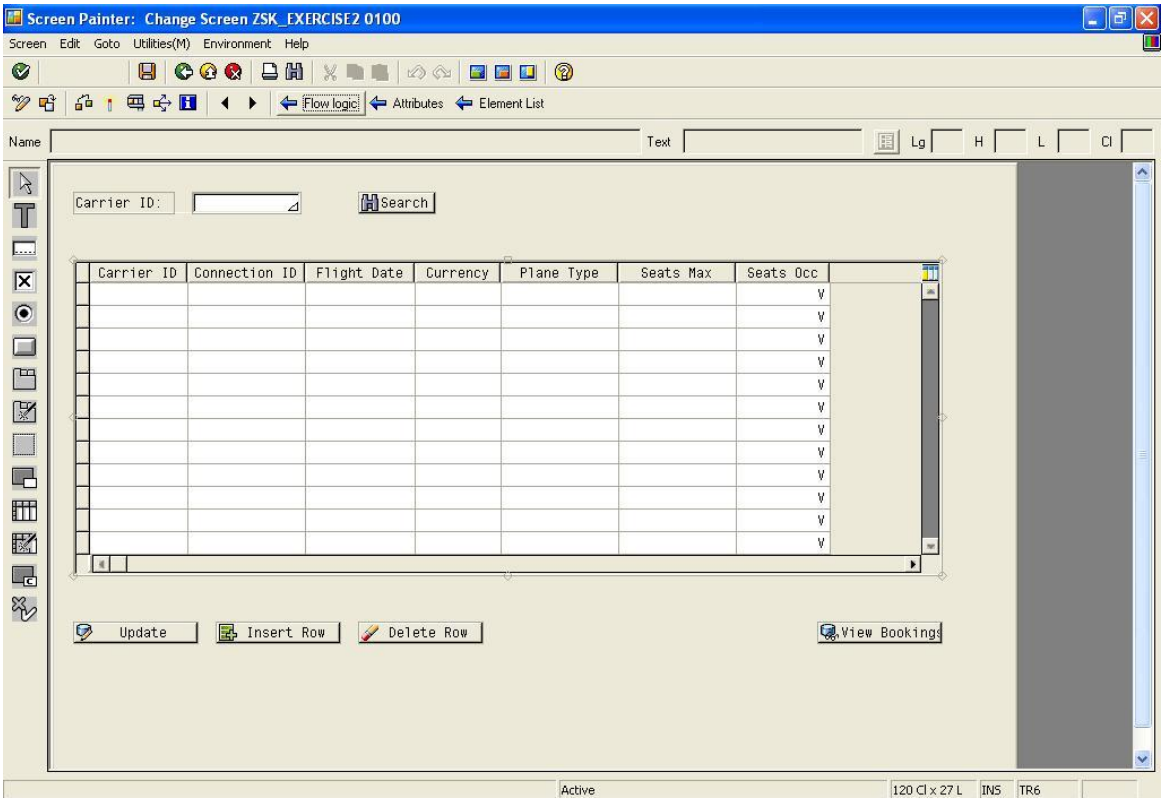

Figure 21 – Screen Painter for screen 100

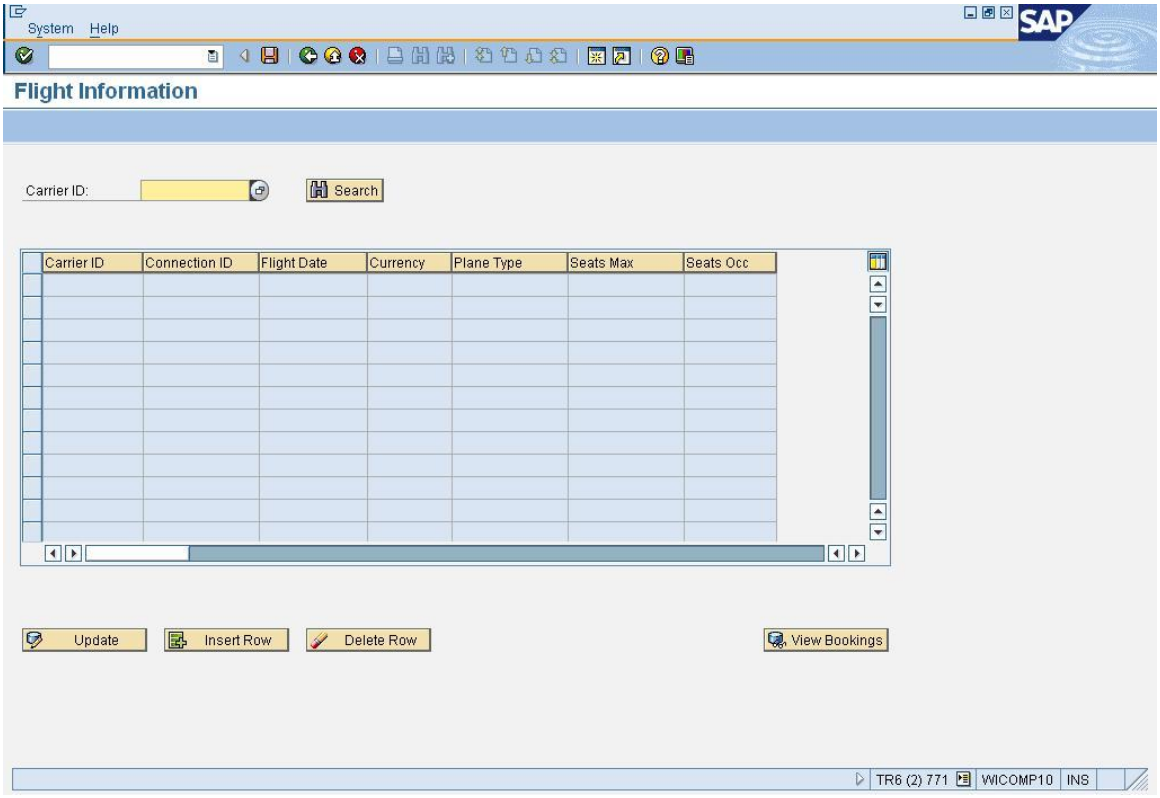

Figure 22 – Output Screen 1

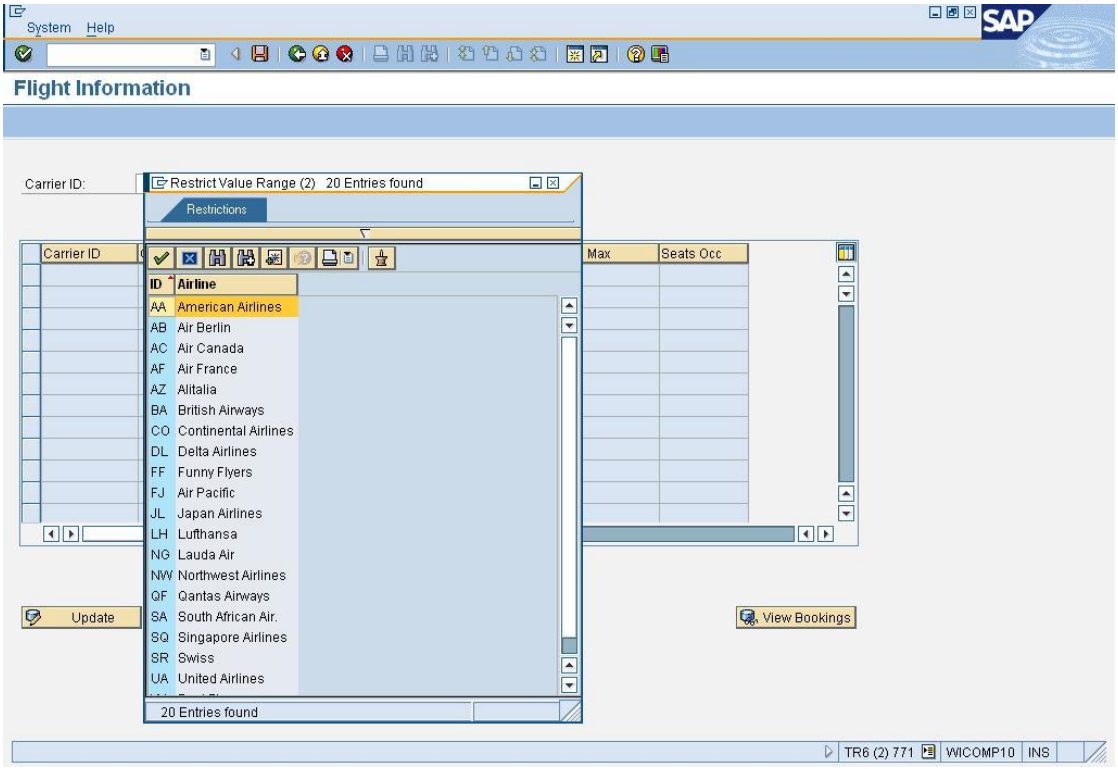

Figure 23 – Entering a 'Carrier ID' to display information

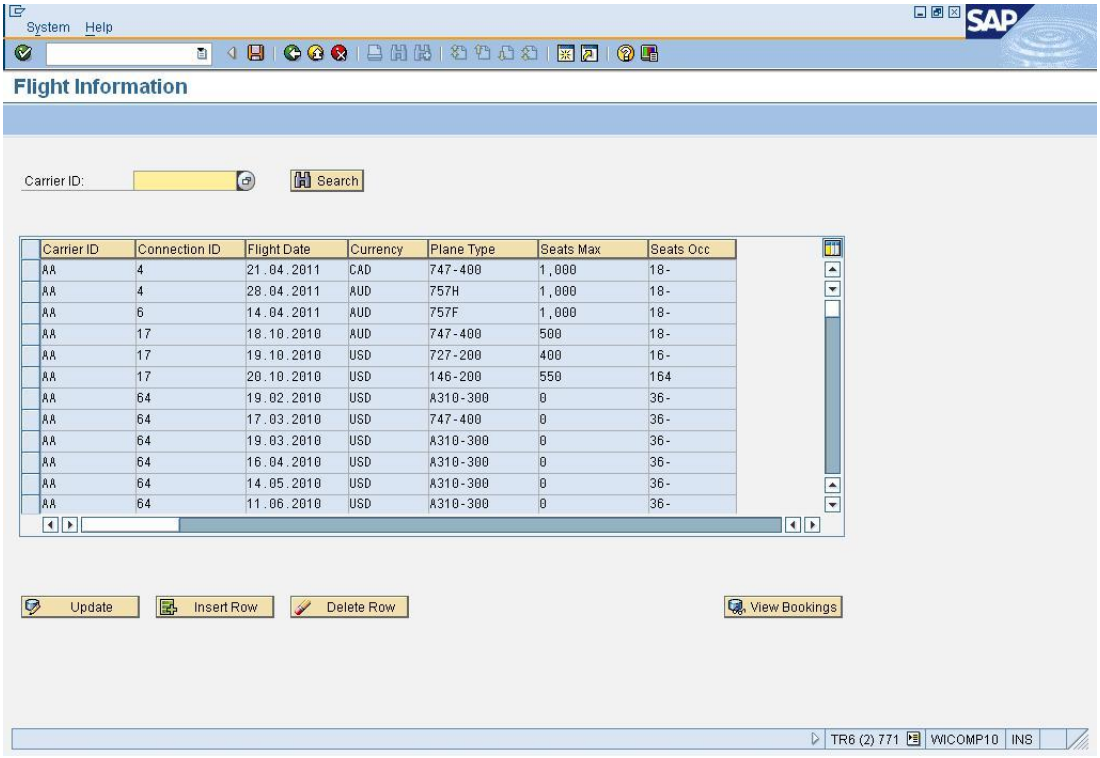

Figure 24 – Relevant information in database shows in table control

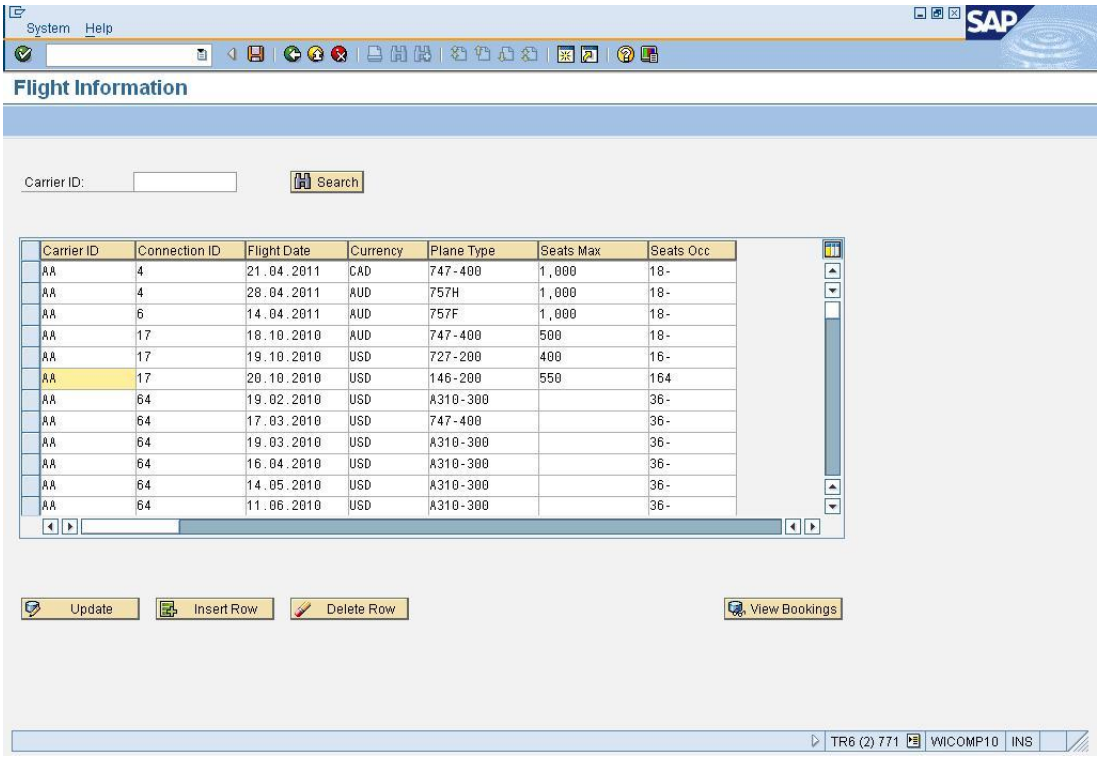

Figure 25 – On clicking 'Update' entire table becomes enabled allowing user to make changes

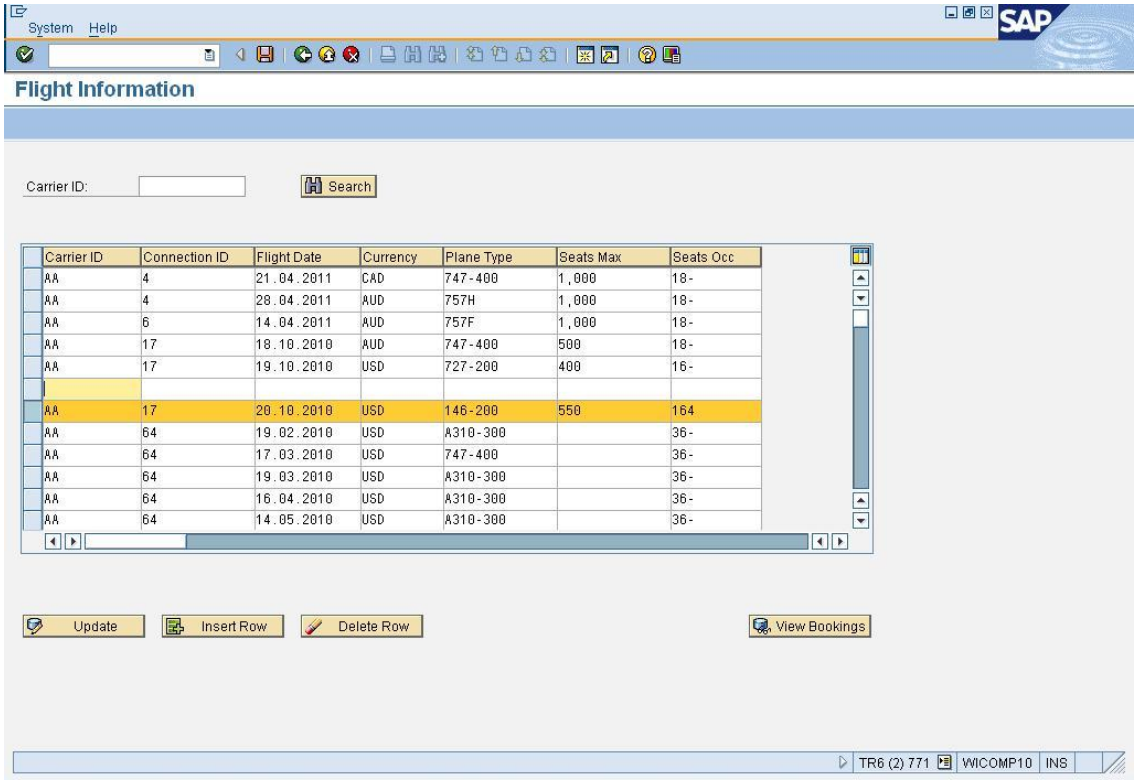

Figure 26 – Select a row and click 'Insert Row' to create a new row above the selected one

Similarly you can also delete a row by first selecting it using the selection column on the left of the table control and then clicking on 'Delete Row'.

All these updates are only made to the table control and the internal table till now. To commit these changes to the database click on 'Save' icon on the top and a message will confirm that all changes made to data using the table control have been saved.

Now to view booking details from the standard table SBOOK according to the 'Carrier ID' and the 'Connection ID', select a row and then click on 'View Bookings'. You will be redirected to another screen with the list of people who have a booking in the selected flight.

User can go back to the previous screen by clicking on the 'Back' button.

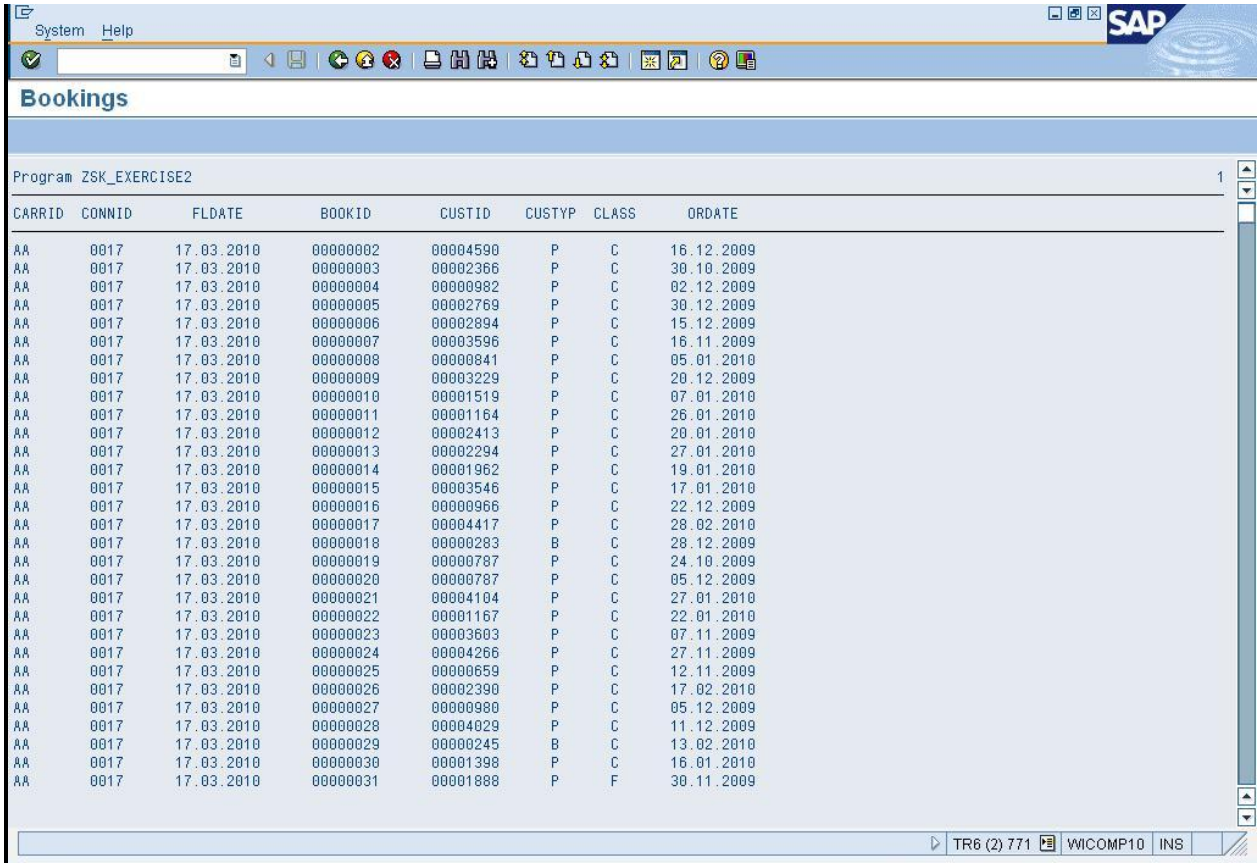

Figure 27 – Booking details for selected flight displayed as a list on screen 101