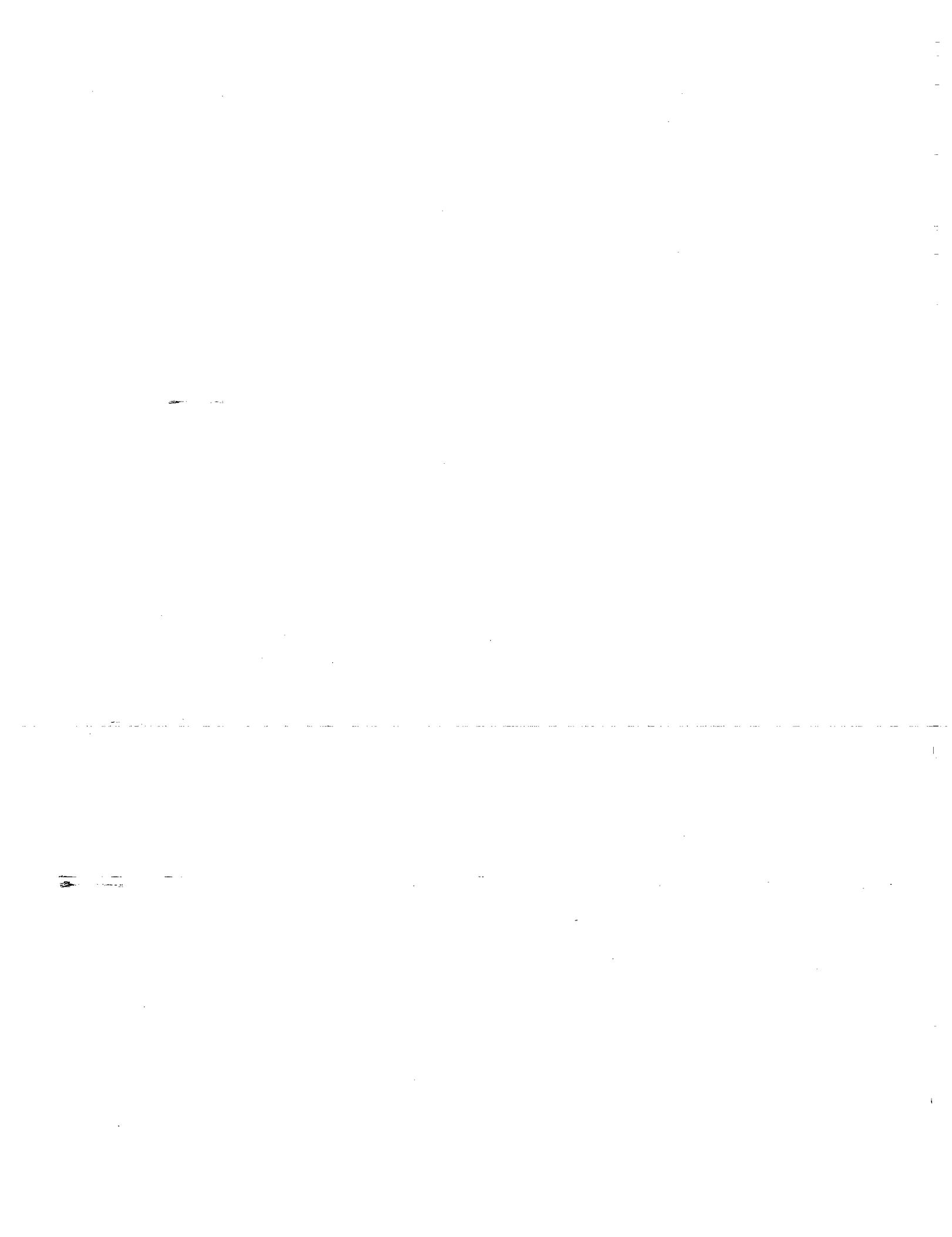


# **People soft**

**NAVYA TECHNOLOGIES**

**FACULTIY: SANDEEP**

**SRINIVAS**



Dhanendra  
K. KARNA

EDITION

⇒ PIA

⇒ Application Designer

⇒ Security 60

⇒ PS Query 66

⇒ PeopleCode 81

⇒ Application Enginee 124

Scribd

D. Satyanareyana.

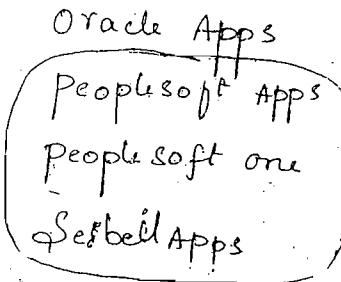
بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

رَبِّ الْعَالَمِينَ

سُبْحَانَ رَبِّ الْعَالَمِينَ

→ People Soft is an ERP Package

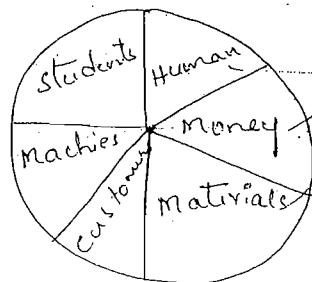
→ ERP — SAP



Now-a-days all these come under oracle Apps.

→ ERP — Enterprise Resource Planning

Eg:



These are all the resources which are needed for an organisation

→ SAP Packages — HR, FI, S&D, CRM

Oracle Apps " — HRMS, Financials, MFG, CRM for customers

PeopleSoft Apps " — HCM, Financials, supply chain

PeopleSoft One " — CRM, EPM, Campus Solution

Each package represents one resource for students

→ A company can choose diff. packages from

diff. vendors b'coz they can be integrated b'coz it is costlier.

Medium, Small companies uses all packages

from single vendor

②

→ Vanilla Implementation : To change the business process according to the ERP (upgrade is easier) without any customisation.

We should not change the ERP.

## People Soft

Technical

↳ people tools

Dev 2000 / JAVA / HTML

ABAP

Functional

↳ people soft Apps

oracle Apps

SAP APPS

people Tools

8.48 / 8.49 → Latest versions ← HCMs 9.0

Financials cms 9.0

CRM 9.0

people soft Apps

To deploy app 3 things are required

① APPS → people tools / people soft Apps

② RDBMS → oracle / DB2 / SQL Server / Sybase  
Informix

③ operating → windows / unix / OS 390  
sys

load tools first on that we have to deploy the app.

- People soft is portable.
- People soft gives some valid combinations of RDBMS & OS. We cannot use as we like.

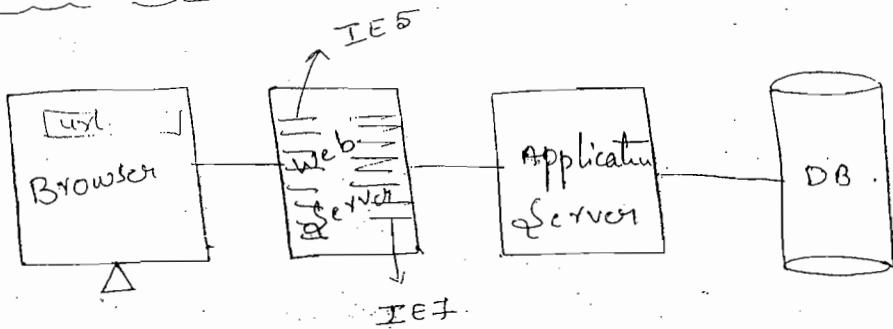
### People Tools

Dev 2000 / VBL .NET / JAVA

- ① Definitions will be stored in Database
- ② Using people tool def. we are going to build Database objects.
- ③ People soft pure internal Architecture — Arch. for web applications
- ④ Definitions (menu, forms) will be stored on File server
- ⑤ Using database objects we are going to develop front end definitions (Forms)
- ⑥ Normal web Architecture

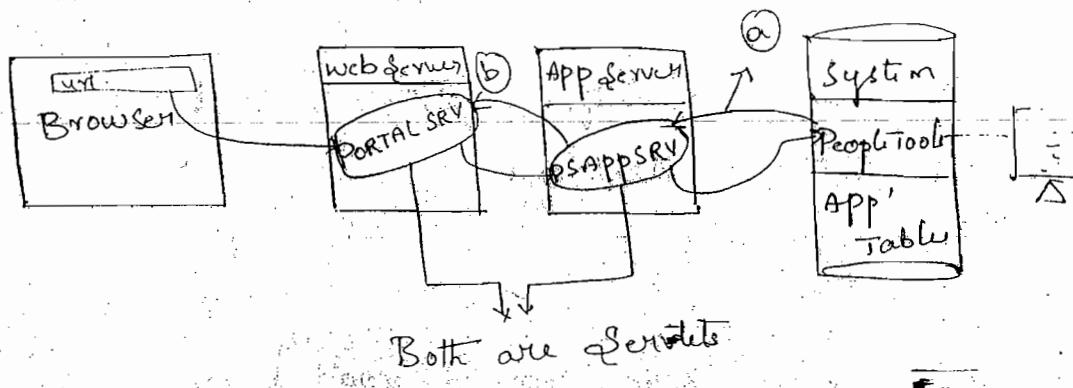
- Technology side SAP is good b'coz of the processor.
- Compared to people soft, SAP is effective while doing online transactions.

## Normal web Architecture



- ① web Developer develops some web pages which are placed on web server
- ② Webserver Maintenance
- ③ changes made in development environment will not reflect automatically in Browser

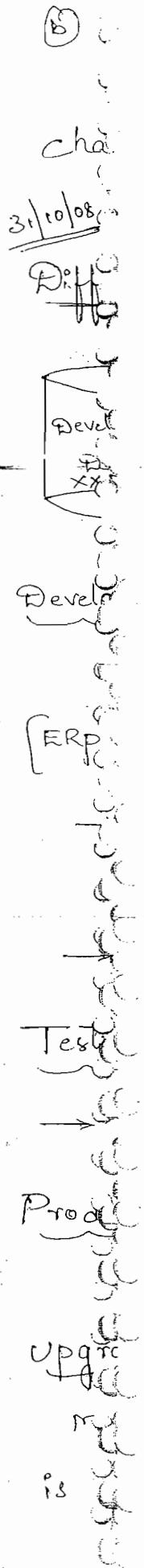
## People Soft Architecture



- ④ Based on the information, PS APPSRV generates

HTML

→ web developers are not req. bcz web pages will be generated automatically by system, so no maintenance is required also.

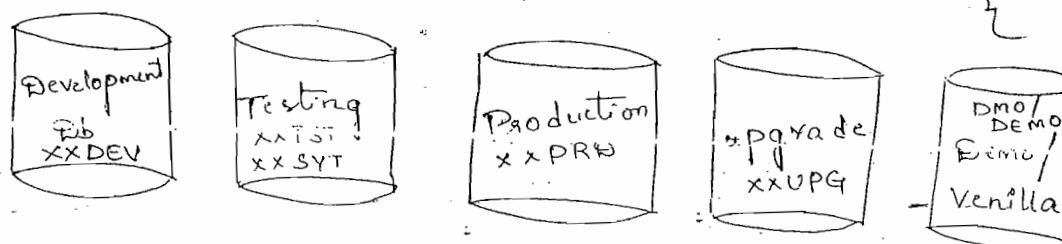


& generates these two.

changes also directly reflect in Browser.

31/10/08

### Different Databases / Instances



Development Db: place where new PeopleSoft object are created.

[ERP Package : Contains pre-defined objects]

→ place where existing PeopleSoft objects are modified.

→ Developers works

Testing Db: place where testing takes place.

→ Testers / QA (Quality Assurance) team works

Production Db: place where clients or end users will work.

Upgrade Db: used for upgradation projects.

Moving from lower version to higher version is called upgradation.

Demo Db: used as a reference database for people  
Soft delivered objects:

→ These are all the diff db which we will see  
in a company.

→ Here we have only 2 db.

- 1) HR
- 2) Finance

### Project Types:

- 1) Implementation (or) customisation project
- 2) Upgrade project
- 3) Support / Maintenance project

### Project Roles:

#### No. of Employees

Ideal case      Generally in a company

1) Technical	—	3	—	5
2) Functional	—	2	—	1
3) Techno Functional	—	2	—	2
4) System Administrator	—	1	—	—
5) Security Administrator	—	1	—	—
6) DBA	—	1	—	1

1) Creates new pages, person who works on people tools.

2) Person who knows the functionality of application

### Flags of Data

functionality of app!

(4)

- will see
- 4) Person who works on how people soft installat takes care of web server, Appserver & Batch server.
  - 5) Person who maintains security.
  - 6) Person who takes care of Peoplesoft Database.

Different windows:

1) Application Designer

Start → programs → PeopleSoft < >



Application Designer

This opens a login page.

User ID :

PS } YP1 }

Password :

PS } YP1 }



For HRMS

For Finance

2) Configuration Manager

Start → programs → PeopleSoft < >



Configuration Manager

3) Query Analyser (Back end)

Start → programs → Microsoft SQL Server

Query Analyser

User ID : sa

Password : sa

4) End user window (or) client window (or) Portal window

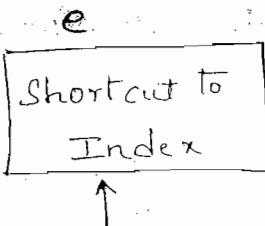
open Internet Explorer

User ID: PS { VP1 }

Password: PS [ ] VP1 [ ]

For HRMS For Finance

5) People Books [Help Documentation]



PSDB - Nothing but collection of tables

→ people soft system supports oracle db, microsoft SQL server, Sybase, Informix, DB2

→ Tables of PSDB are mainly divided into 3 parts

1) System catalog Tables

2) People Tool Tables

3) Application Tables

 <b>System catalog Tables</b>	<ul style="list-style-type: none"> <li>Tables created during db installation</li> <li>Name is prefixed by 'sys'</li> <li>Eg: sysObjects, sysIndex etc</li> <li>DBA works on these tables</li> </ul>
<b>People Tool Tables</b>	<ul style="list-style-type: none"> <li>Tables created during people tools installation</li> <li>Name is prefixed by 'ps'</li> <li>Eg: PSRECDEFN, PSDBFTLD etc</li> <li>These are meta data (Data about Data) tables</li> <li>These store info. about people objects</li> <li>System administrator will work on these tables</li> </ul>
<b>Application Tables</b>	<ul style="list-style-type: none"> <li>Tables created during app. installation</li> <li>Name is prefixed by 'ps_'</li> <li>Eg: PS_JOB, PS_Account etc.</li> <li>Store App' Data</li> <li>Technical &amp; functional people works on these tables</li> </ul>
a) control Tables  b) Transaction Tables  c) Run control Tables.	

### Control Tables :

- used to store control information:
- 'SETID' will be high level key field (First field)

in the table as well as key field)

→ This data is rarely operated.



3/11/08

① Dev

→ Using  
or  
soft

① App

② App

③ people

④ File

⑤ comp

⑥ Int

⑦ Work

⑧ Glob

Develop

→ App

Obj

→ App

### Transaction Tables:

→ used to store transaction information

→ 'BUSINESS-UNIT' will be high level key field

→ This data is frequently operated

### Runcontrol Tables:

→ used to store run control information

→ 'OPRID' & 'RUN-CNTL-ID' will be high level key field

↓            ↓

operator ID      Run control id.

### Advantages:

→ used to run any report or process

→ used to provide input parameters (or) run

control parameters.

Develop

→ App

Obj

→ App

## ① Development Tools

→ Used to create or modify people soft objects.

- 1) App' Designer
- 2) App' Engine
- 3) people code
- 4) File layout
- 5) component interface
- 6) Integration Broker
- 7) work flow
- 8) Globalization

## Development Tools:

→ App' Designer is used to create/modify people soft objects. This is basic tool.

→ App' Engine is used to perform background SQL processing.

## ② Administration Tools

→ Used to maintain or administer security, servers & Data.

- a) Security administration tools
  - i) Menu level security
  - ii) Definition ..
  - iii) Portal ..
- b) Data administration tools
  - i) Data Mover
  - ii) cube manager
  - iii) Data integrity
  - iv) Achieve Data
- c) Server administration tools
  - i) PS Admin
  - ii) PIA

## ③ Reporting Tool

→ Used to create, modify reports

- 1) Tree manager
- 2) peoplesoft Query Manager
- 3) PS nvision

## 3<sup>rd</sup> party tools

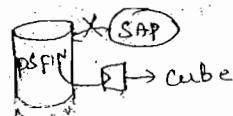
- 1) SQR
- 2) Crystal report

Does not belong to  
peoplesoft system

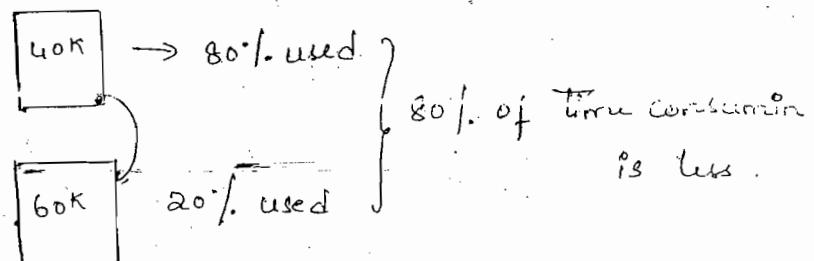
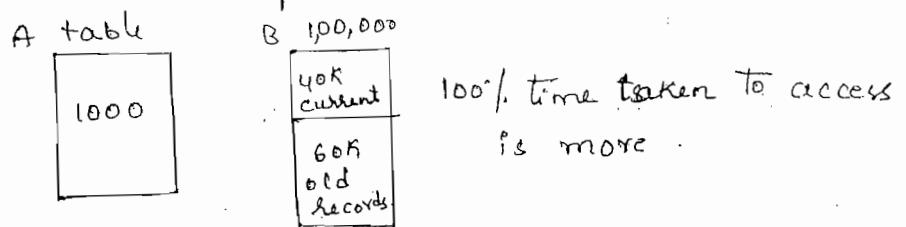
- SQR is not accessed by clients, whereas app' engine can be " " "
- people code is used to implement business logic or rules.
- file layout
- Component Interface      Integration tools  
Integration Broker      we can share data b/w systems
- workflow used to automate time consuming approval systems.
- Globalization means supporting multicurrencies & multi languages.

### Administration Tools

- Menu level security is used to provide page access.
- Obj. security is used to secure people soft def's or objects.
- portal " " " " portal objects.
- Data mover " " " move data from one db to other db.
- cube manager " " " share data space b/w systems  
nothing but aggregation of data



→ Archive data is used to take file backup to increase search speed.



→ ps Admin is used to maintain webserver & application server.

→ PIA (people soft internet architecture)

### Reporting Tools:

→ Tree Manager is used to maintain data in hierarchy.

→ used People soft Query Manager is used to generate select statements by performing GUI operation.

→ psenvision used to integrate Excel with psDB.

→ cube Manager & Archive data are rarely used

→ ps Admin is imp. for sys. administrator.

## Application Designer :

→ Used to create/modify people soft def or objects.

Start

↳ programs

↳ peoplesoft < >

↳ App' Designer

PS } FOR VPI } FOR  
PS } HRMS VPI } finance

→ Using this we can create

- 1) field : column in a table/record.
  - 2) Record : collection of fields.
  - 3) page : collection of records
  - 4) component : collection of pages
  - 5) menu : collection of components
  - 6) SQL Definition : used to store SQL statements
  - 7) style sheet : used to change appearance of front end.
  - 8) project : collection of other peoplesoft objects
  - 9) HTML : used to store HTML coding
  - 10) File Reference : used to store file directory
  - 11) App' engine : used to perform Background SQL processing
  - 12) File Layout
  - 13) component interface
- } Integration Tools

- 14)
- 15)
- 16)
- 17)
- 18)
- 19)
- 20)
- 21)
- 22)
- 23)
- 24)

(E)

Field

Def

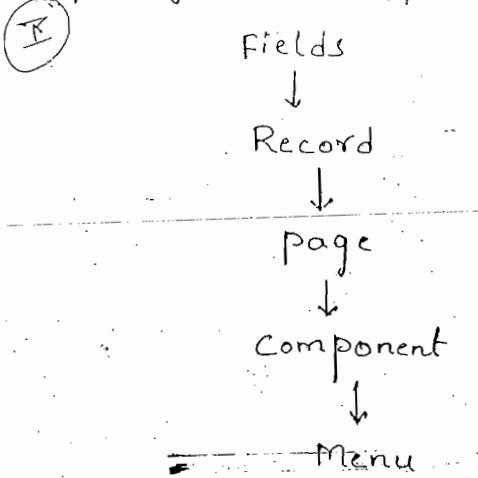
1. col

2. col

- Projects.
- 15) Business process } Workflow Tool
  - 16) Approval Ruleset }
  - 17) message }
  - 18) Business Interlinks }
  - 19) message channel } Integration
  - 20) optimization model }
  - 21) mobile page }
  - 22) problem Type }
  - 23) APP' package : used to store people code ;
  - 24) Image : used to store images .



Imp. objects in APP' Designer are



Fields :

Different field Types .

1. Character :

Max. length 256 .

2. Long character:

length > 256.

of type

①

1)

2)

3)

4)

5)

6)

7)

8)

3. Date:

Length = 10      MM/DD/YYYY  
12 34 56 78  
↓ ↓  
9 10.

4. Time:

Length = 13      HH : MM : SS : mmmm  
12 34 56 78 9 10  
↓ ↓  
11 12 13

5. DateTime:

Length = 23

6. Number:

Natural numbers 0 - ∞

+ve decimal numbers

7. Signed Number:

Integers - ∞ - 0 + ∞

+ve & -ve decimal numbers

8. Image:

Used to store Images

9. Image Reference:

Used to store path of the image

10. Attachment:

Used to store document / excel files

(1)

Open App' Designer

- 1) File → New (or)  $\text{ctrl} + \text{N}$  or
- 2) Select field from list.
- 3) Select the field type.
- 4) Specify length
- 5) Specify label
- 6) Select the field format (default is upper case)
- 7) Save the field

File → Save (or)  $\text{ctrl} + \text{s}$  (or)  (floppy button)

- 8) Specify field Name

App' Designer - untitled → Title Bar

File Edit View → Menu Bar



→ Tool Bar

untitled

project  
space

work space /

Development space

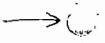
upgrade

Develop

Log window / o/p window

Build Upgrade Results Validate

4/11/08



Select New file.

① Select field

Field Type: character (Default)

Field Length: 40

Field Labels:

	<u>label id</u>	<u>long name</u>	<u>shortname</u>	<u>Def.</u>
1.	Ename	Employee Name	Name	<input checked="" type="checkbox"/>
2.	Dname	Dependent Name	Name	<input type="checkbox"/>
3.	Cname	Customer Name	Name	<input type="checkbox"/>

Default  
Name  
↑

Field format

Format type: uppercase (Default)

Family name:

Display name

(2)

Give the field with name B 25 character.

② Field Type: Number  signed.

can store only +ve &  
the numbers negative numbers.

Integer positions: 7 Decimal positions: 2

1. Salary Total Salary Salary

① open the reg. field (File → open)

Definition : Field

Name : B-25



Displays all B-25 tables.

Select B-25 character.

Next Insert

↳ current Def. into project (F7)



This Displays def. in project space.

Fields

↳ B25-character

② Go to menu

↳ Insert

↳ Def. into project (ctrl+F7).

Def. Type : fields

Name : B25

Select B25 Number

click on **Insert** button

any

In this method no. of fields can be inserted

into project at a time by just double click  
on it.

Save project as B25\_Fields-project

File → Save all.

### Translate Fields:

→ These are nothing but normal fields, which have collection of accepted values entered.

↳ Translate values.

Empno    name    status

Active  
Terminated  
inactive  
Absent

fields

Accepted  
values.

xyz.

### I Conditions:

- 1) The data type of field must be character.
  - 2) Length should be less than or equal to 4 character.
- For better performance we can have up to 40 translate values.
- Fields which are used to store collection of accepted values.
- If the translate values are exceeding 40 instead of using translate fields we use prompt tables.

↳ New

↳ Select Field.

Field Type : character

" length : 3

1. status Employeestatus status

Same → B25 - Trans-field.

To add translate values we have to enter property

For this click on properties icon

Field properties.

click on Translate values.

1. ABS  Absent

Add

2. ACT  Active

3. RET  Retired

4. TER  Terminated

→ If we want to inactivate the terminated value.

Go to properties

↳ Translate values

↳ Double click on terminated

inactive

OK

→ Translate values tab will appear, if we create the field only with the 2 cond's specified.

→ properties window

field properties

General

B25-character

Field Definition

1) used to specify descriptions  
2) used to capture company commenting status

Owner's Id: [dropdown menu]

Used to restrict this def. to a specific module

Last updated:

Date/Time : 11/04/2008 8:05:00AM

By user : PS

OK Cancel

People

Field

PS

PSQ

PSY

Y

→ Ge

G

C

C

C

→ PI

da

Rec

C

C

Tar

Via

C

31/10/08

Type

DSQ

SC

Field def. information is stored in these tables

PSDBFIELD : Field definition table.

PSDBFLDLABL : Field label table.

PSXLATITEM : Translate value table.

→ Go to Query Analyser

Select \* from PSDBFIELD

where FieldName = 'B25\_character'.

→ If we want to do any modifications we can do it directly.

Records :

Collection of fields

Table : can store data

View : cannot store data

uses : 1) used to get data from multiple table

2) To get a limited set of data.

Types: II

1) SQL Table : Used to store data, collection of field

2) SQL View : " " get data from multiple table  
→ collection of fields & select stmts

→ Select stmt is manually written.

→ Build process is req. both for SQL Table & SQL View.

### 3. Query View:

5.) S

→ Used to get data from multiple fields, tables.

→ coll. of fields & Select statement.

→ Select stmt is generated using PS Query or PeopleSoft Query Manager.

→ Build process is required.

### 4. Dynamic View:

→ used to get data from multiple tables

→ collection of fields & SELECT stmt.

→ Select statement is manually written.

→ Build process is not required.

→ SQL view & Query View are database objects, so build process is req. whereas Dynamic View is not db object.

→ SQL view & Query View are database objects, so SQL structure is stored in db. If we want to see it in front end, it get directly from db.]

→ Dynamic view also called as runtime view.

→ we can have meta SQL stmts (Database independent SQL stmts).

S.

C.

D.

E.

F.

G.

H.

I.

J.

K.

L.

M.

N.

O.

P.

Q.

R.

S.

T.

U.

V.

W.

X.

Y.

Z.

6. D.

E.

F.

G.

H.

I.

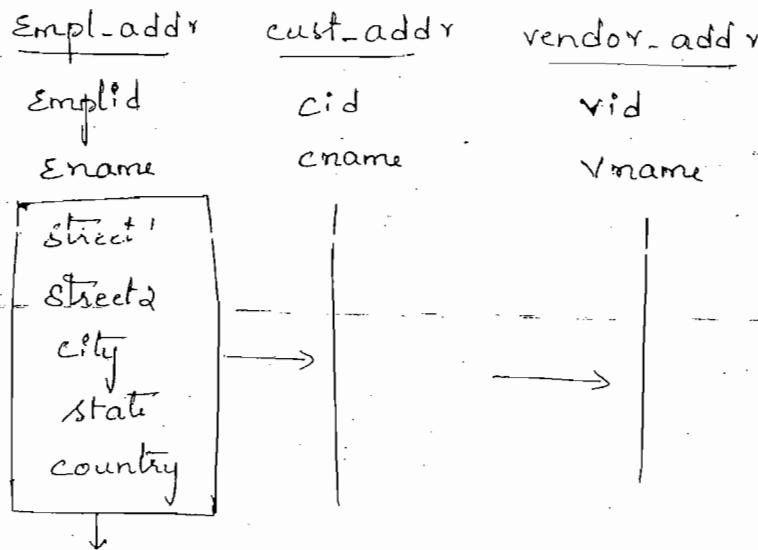
J.

K.

L.

### 5.) Sub Record:

Collection of fields that can be used in other records.



These fields are common so we create a record with these fields called as sub-record.

→ Used for reusability & easy maintenance.

→ Not a db object so business build process is not req.

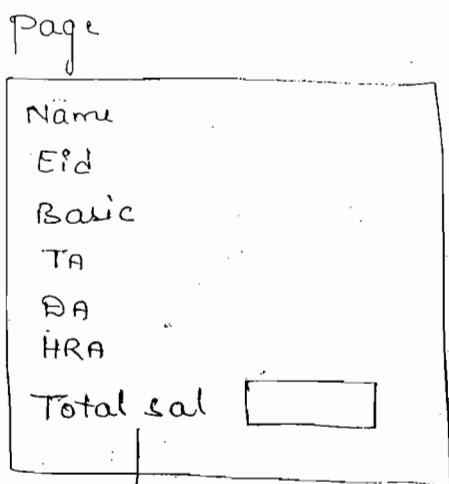
### 6. Derived / Work record:

→ Collection of fields.

→ Used for online page calculations.

on the page, with the data available we perform some operation & displays it.

Eg:



This is calc. with data available on page.

→ Used as function Libraries

→ Build process is not required.

#### 7. Temporary Table:

→ collection of fields.

→ used to increase performance of App' Engine programs

→ Db object so build process is req.

#### 8. Steps to create a Record:

1. open App' Designer

2. File → New (or)  $ctrl+N$  (or)

3. Select Record

4. Insert fields

a) Insert → Fields

b)

5) qe

a)

b)

6) qe

7) qe

8) qe

9) qe

Eg:

File

Go

It

Ca

Cl

Cr

Re

Rd

Re

Re

Re

5) Select key fields.

a) Double click on the reg. field.

b) Select Key.

Select record type.

6) Save the record.

File → Save (or)  $\text{ctrl} + \text{s}$  (or) 

7) Specify record name.

8) Design the reg. Select stmts & save

(SQL View / Query View / Dynamic View)

9) Build the record.

(SQL Table / SQL View / Query View / Temporary View)

Ex:

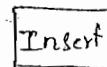
File → New

↳ Select Record.

Go to Insert menu.

↳ Select Field.

Name: Emplid



Select the reg. field.



Like wise select Name & Salary / monthly rent.

Double click on Emplid

↳ Select Key







Click on Record Type tab

① SQL Table

Save it with B25-SQL-Table



To Build the record

Build → current Definition  
(or)

Three coloured button. (click)

Build

B25-SQL-Table	<input type="button" value="Build"/>
	<input type="button" value="cancel"/>
	<input type="button" value="Settings"/>

Built options:

create Table : used for SQL Table

create Indexes : used for key fields

create View : used for SQL View

Alter Table : modifying existing SQL Table / Temporary Table

Create Trigger : used by DBA

① Build script file : used to create a file with req. SQL statements.

② Execute SQL Now : Executes the stmts available in the script file.

③ Execute & build script : create & execute the script file.

→ Go to Query Analyzer.

If the table does not exist.

Mark the foll.

Create Table

Create Indexes.

Execute & build script

Open the SQL build script process file in Notepad  
log file  
to see what stmts it prepared.

Drop: Deleting the data & also structure of the table from db.

PBuild.sql → is the file consists of SQL stmts prepared.

PBuild.log → consists of error & success messages.

SQL View

Fields + Select Stmt

Run

File → New

↳ Record



Edit

Insert → Fields

↳ Empid, Birthdate, Birthcountry.

Help

click on Record Type Tab

↳ ① SQL view

click on Save B25 - SQL View

File

click to open SQL editor



This

Select Emplid, Birthdate, Birthcountry

From ps\_Person



Build → current def

create views

① Build Script & execute

To insert into project

Insert → current def. into project



This inserts view as well SQL statements

File

into project.

Edit

Help

File

Edit

Help

File → New

↳ Record



Insert → Fields

↳ Empid, Birthdate, Birthcountry

Record Type Tab

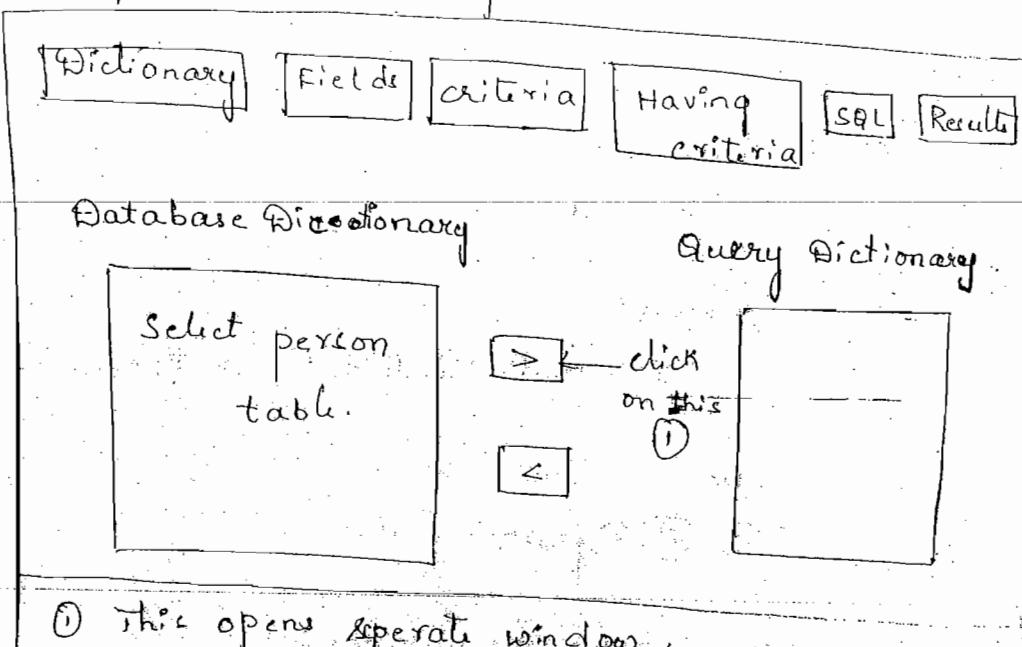
↳ ⚡ Query View

Save B25-Query-View

Record Type

↳ click to launch query

This opens a window for us



① This opens separate windows

Select the fields req.

Emplid → col. NO 1

↳ (Double click)

Birth date → col NO : 2

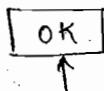
Birth country → col NO : 3.

click on SQL Table.

we can see what is the select statement

Query manager is prepared.

To see op click on Results Tab.



Build →

create view

executing Build SQL

Dynamic View:

File → New

↳ Record.

Insert → Fields

↳ Emplid, Birthdate, Birthcountry

Record type

↳  Dynamic view

Save BAS-Dynamic-View

Record type

↳ click on SQL Editor

PS - Person

we cannot build this dynamic view.

→ This is used for pages.

Derived / work Record:

File → New

↳ Record

Insert → Fields

↳ Empid, Name, monthly-Rt

Record Type

↳ Derived / work Record ①

Save Bas-Derived-work.

→ we cannot build

→ used for online page calculations, used for  
func. libraries

Sub Record:

File → New

↳ Record

Insert → Fields

↳ Address1, Address2, city, state,  
country

Record Type

↳ ① Sub Record

Save B25\_subRecord

→ we cannot build this, so structure is not def. in db (backend).

Eg:

File → New

↳ Record.

Insert → Fields

↳ Emplid, Name

To insert sub record.

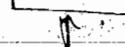
Insert

↳ Subrecord



Name : B25\_subRecord

Insert



Double click on Emplid



Key



Record Type

↳  SQL Table

Save B25\_subTable

Build →  Create Table

Build & execute

To

C

Tem

→

Conf

1. PR

2. T

Fl

→

H

→

D

→

R

→

Sa

→

Bu

→

Q

→

G

→

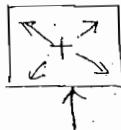
F

→

T

→

click on expand sub records



Temporary Table :

Conditions :

1. "PROCESS\_INSTANCE" must be high level key field
2. The name must be postfixed by \_TAO/\_TM

File → New  
↳ Record.

Insert → Fields

↳ Process-Instance, Emplid, Name

Double click on " " & select key.

Record Type  
↳ Temporary Table.

Save B25-Temp-TAO

Build →  create table

→ ① Build & execute.

Query Analyser.

↳ Select \* from p-B25-Temp-TAO

## Modifications for a Table

### D) Adding a New Field:

B25\_SQL-Table.

Insert

↳ Fields

↳ DOJ, country

Save it. If we see in db, DOJ field will not appear, we have to build after modification

Build →

Alter Table

① Execute & build.

PSBuild.sql:

we have table

PS-B25-SQL-Table Add country

PSYB25-SQL-Table

moving  
data to a  
dummy table.

↳ Emplid

Name

Monthly\_Rt

DOJ

Country =

↓  
Assigning " " to country

Next Dropping PS-B25-SQL-Table

Now renaming dummy table

PSyB25-SQL-Table

↳ PS-B25-SQL-Table

## Settings

Build settings.

### Create

Table creation options.

Recreate table if it already exists

Skip table if it already exists

Select this skip table for app'.

View creation options

Recreate view if it already exists

Skip view if it already exists

Index creation options

Recreate index if it already exists

Recreate index only if modified

### Alter

Drop column options

skip record

Change col length options

Truncate data if field too short.

Skip record if field too short.

Alter Any

Add fields

changes field

lengths

Renames Table

Delete fields

Alter even if no changes

ALTER Table options

{ O Alter in place.

{ O Alter by Table Rename.

Scripts

Logging.

These are used to specify where to create script file & log files.

Properties:

Record properties are classified into

a) Record properties

To Access

open Record.

↳ click on properties icon (8) Alt + L

b) Record Field properties

To Access

open Record.

↳ Double click on Field.

Rec

Box

General Tab.

Info

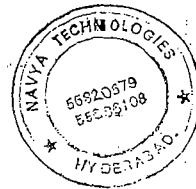
# Window

General

use



B25\_SQL\_Table



Description : Short Description

Record Definition :

- Long Descriptions
- Company commenting standards

General Tab

Owner ID : Restrict to specific module

Last updated : Generally we keep this

Date / Time : blank to access

By user : ps

Globally,

OK

cancel

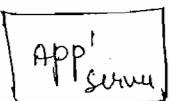
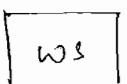
11/10/08

use Tab

Set control field :

 A

This is used to retrieve data as a set



Did	Eid	Name	Sal.

50 records

If we set  $Did = 10$  then it will bring all the Did belongs to 10 at a stretch.

### Record Relationships:

Parent Record: used to maintain parent child relationship.

### Parent child relationship:

process of breaking a table into small tables is

Called Normalisation.

1) used to reduce data redundancy

2) effectively utilize database space

Did	Dname	Loc	Emplid	Name	Sal.
10	comp	Hyd	101	A	10K
10	comp	Hyd	102	B	20K
10	Comp	Hyd	103	C	30K
10	comp	Hyd	104	D	40K
20	elec	Bang	201	E	50K
20	elec	Bang	202	F	60K

Tot.

T.

Di.

key 1c

2c

3c

T.

Conc

1)

2)

Rel

Eg:

A

B

C

D

E

F

TBL1

ID	Name	Loc
key 10	comp	Hyd
20	Elec	Bang

$3 \times 2 = 6$   
PR

TBL2

ID	Empno	Name	Salary
10	101	A	10K
10	102	B	20K
10	103	C	30K
10	104	D	40K
20	201	E	50K
20	202	F	60K

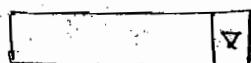
CR  
 $6 \times 4 = 24$

$$\text{Total cells} = 6 + 24 = 30$$

Conditions to maintain Parent child relationship:

- 1) child record must have all key fields from parent record & atleast one additional key field
- 2) In the child record properties, specify parent record name: ]

Related Lang. Record

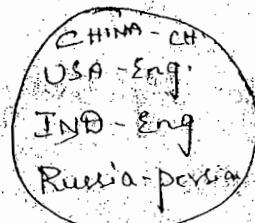


↓

Used to maintain multi languages.

Eq:

A client has business in USA, IND, Russia



emp1-TBL		
EID	Name	Salary
101	Ravi	10K
102	Siva	11K
103	Giri	12K

This table is understood by USA & IND people  
Base Lang. Record

EID	lang_cd	ename	sal
xxx	Per	xxx x	xxx
xx x	Per	xxx xx	xxx
xxx	Per	x x x x	xxx
	CH	xx xx	xx x

Related Lang. Record  
This table is in persian lang. readable by Russian people

→ The table in which we maintain data other than english is called Related lang record.

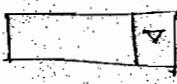
Conditions:

→ 'Language\_cd' should be one of the high level key field.

→ Name must be postfixed by '-LNG'

→ specify the related lang. record name in base lang. record properties

Query Security Record:



Optimization Delete Record:



Used for high level security

99% we don't use this

They will give the values if needed.

(1) A



2 F

①

If

Re

→ C

or

Z

Y

X

T

C

E

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

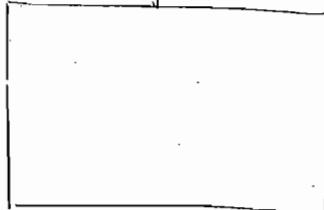
Z

→ Used to monitor confidential/secured info. in a table.

## 2 types of Audit:

### ① Record level Audit:

No fields



If 30 fields are confidential then we go for

Record level Audit.

→ Used when % of fields that need to be monitored is more.

System will monitor the table.

→ The table into which system captures the monitoring info. (or) The table used to store monitoring / Auditing information is called

Audit Table.

→ Record level Audit is captured in user defined Audit records.

### ② Field level Audit:

→ used when % of fields that need to be monitored is less

→ captured in psAudit

① Create a Table (B25-Empl-Sal-Tbl)

EmplId - Key, Search Key

Name - Alternate key

DOJ -

Basic

TA

DA

HRA

② Create a page (B25-Empl-Sal-Pg)

EmplId	<input type="text"/>
Name	<input type="text"/>
DOJ	<input type="text"/>
Basic	<input type="text"/>
TA	<input type="text"/>
DA	<input type="text"/>
HRA	<input type="text"/>

Sal

To

①

②

③

④

⑤

⑥

⑦

⑧

③ Create a Component (B25-Empl-Sal-Cmp)

Search Record: B25-Empl-Sal-Tbl

④ Register the component

↳ New

↳ Project

File → New

↳ Record

Insert → Fields

↳ Emplid, Name, DOJ, Basic, TA, DA, H

Key

Alternate key

Search key

— key —

Record Type

↳ SQL Table

Save B25-empl-sal-Tbl, Build it.

Insert Record into project

Save Record project B25-Audit-project

To create a page:

① File → New

② ↳ page

③ From project space Drag & Drop req. fields from

Records. (or) Drag & Drop complete record.

Use Mouse to alignment.

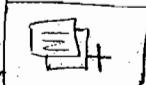
④ Save it B25-empl-salPg

⑤ Insert → current def. into project

## To create a component :

- ① File → New
- ② Select component 
- ③ Drag & Drop the page from project space
- ④ properties → Search Record : B25-Empl-sal-Tbl
- ⑤ Save it      This is compulsory, unless  
 ↓ :      this we cannot save comp.  
 B25-Empl-sal-comp

## Registering component

- ① open the component
  - ② click on Register component button 
- This opens a separate window.
- ③ 1<sup>st</sup> page is start page
- Add this component to a menu
  - Add this component to a portal registry
  - Add this component to a permission list

Next

- ④ Add to Menu & bar page

Menu Name :

Select

Bc

f

c

✓

✓

✓

✓

✓

✓

✓

✓

✓

✓

✓

✓

✓

d

✓

✓

specify any Menu & bar Name

Next

Empl-sal-project

Empl-addr-project

② create content page

Ref.

① Target content    ② Homepage pagelet

✓ Portal Name : Employee    [Select]

✓ Folder Name : myFavorites    [Select]

Content Ref. Name : Empl-sal-cmpt-gbl

Content Ref. Name : Empl-sal-cmpt-gbl

Long Description  
[254 chars] : Empl-sal-cmpt-gbl

Sequence Number : 10    product : PT

✓ Template Name : Default-Template [V]

Object owner Id : [ ] [V]

Always use default local node.

Next

③ Add to permission list page

permission list Name : AEAE1000    [Select]

Next

e) Finish page



Wizard Game complete

↑ means comp. reg. completed successfully.

Folders can have sub folders or content references.

microsoft IE

↓  
my favorite

↳ B25 - Emploal - cmpt:

↓  
opens a search page

click on

emplid

Add the data &

~~01/08~~  
Steps for Record level Audit : Emploal project

i) Create Audit record

ii) The foll. fields should be high level key fields

i) Audit = OPRID

ii) Audit = STAMP

iii) Audit = ACTN

performing the data operation.

Audit-Stamp: Used to capture the date & time of data operation.

Audit-ACTN: Used to capture what data operation is being performed.

- b) Name must be post fixed by -ART / -AUDIT
- 2) For 'AUDIT-STAMP' select Auto update & system maintained Record field properties.
- 3) Specify Audit record name in actual table properties & select the data operations that need to be monitored.

Eg:

File → New

↳ Record.

Insert → Fields

↳ Audit-oprid, Audit-stamp, Audit-Ac

Double click on each of these & select  Key

Drag & Drop fields of B25-EmpSal-Tbl.

Save it with B24-EmpSal-ART

(or)

Open the table → save As B24-EmpSal-ART

Insert → Fields → Audit-oprid,

Double click on AUDIT-STAMP



→ go

to

→ FI

to

System maintained

Auto update.

Save

Build → create table

Build

Go to B24-EmpSal-Tbl.

Properties

\* In audit table we should  
not keep any key values.

Use Table

↓

Record Name : B24-EmpSal-ADT

change

→ Go to IE → my Favourites → our component

↓  
change the data

& then see the table in Query Analyzer

→ Use Tab

Selective

↓

If we select Selective option, it only captures  
the field value that have changed in Audit table.

where RECNAME = 'B25\_EmplSal.Tbl'

→ Field properties of Basic in actual

↓  
table.

Audit:

Field level  
Audit.

Add Field

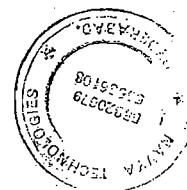
change,

change the data or add the data in the  
Comp. That is registered.

Then go to Query Analyser & check.

Record Field properties:

Use Tab:



Field Name : Emplid

Keys

Key → uniqueness of data, Indexes, Not Null

10)

103 It diff. to find where 225 Emplid; if

646 Key is selected, the data is aligned

356 in ascending order. Searching speed

474 will be increased.

635

225

Not Null → without entering a field value we

can't insert or save the row of data.

Name	sal	age

For this table we can't decide which is key field. No field will be key field. For such type of tables we use Duplicate order key.

### Duplicate Order Key

→ No uniqueness, Indexes, Not Null

### Descending Key

EID	DOJ	Name
Old	101	
New	102	

we see  
100 → 80% new employee  
data  
20% we see  
old employee  
data

we can say whether he is old, New employee is by DOJ field.

If New employees data is at bottom, speed of execution is less.

so if we set DOJ as Descending Key then bottom will be old employees data & at the top New employees data is stored.

∴ Speed is increased.]

→ used for date fields to align data in desired order.

search key → This is used for searching purpose on search page.

value.

on search page

Search by:

we have to select search with exact value.

List Box Item : used when multiple values are displayed from a single field.

For all search key & for alternate search key we have to select list box item.

Alternate Search Key : used to make the search process easy.

From Search Field      } used to search for a  
 Through Search Field      } range of data.

SK	SK	SK	ASK		
F1	F2	F3	F4	F5	F6
			-		

on search page

If we want to see F3 default instead of F1 then select

Default Search Field : used to select field that need to be default appeared on search page.

Disable advance search option:

we can only search by 'begins with' in advance search page.

Normally advance search page contain <, >, =, not equal to along with begins with.

### Audit

- Field Add
  - Field change
  - Field Delete
- } Used for field level Auditing

- Sys. Maintained
  - Auto-update
- } Used for fields where data is maintained by sys.

### Record Field Label ID

A field has no. of labels, e.g.: Emplid, App'Id...

If we select Applicant Id as default then in front end instead of Emplid, we see App'Id

Eg: Applicant Id begins with

Suppose for all employees say 99%, the travel allowance is 800.

Every time we have to enter 800. Instead of

Default value

28

Constant

(or)

Record Name

Field Name

800	*
-----	---

	*
--	---

	▼
--	---

} If the data is present in any record then

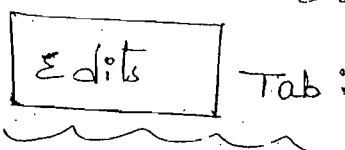
specify these two. A field

Default page control. field has many rows, it populates the first row value.

System Default ▾

Depending on field type sys. automatically set the page control.

Eg: of page controls: Edit box, Radio button, List box etc.



Field Name : BASIC

Required

If we enter data in front end without entering basic sal value, sys. allows it & saved.

But the employee data should not be entered without basic field. Then if we select

Required, it gives error msg while saving without entering basic sal value.

Edit Type :

No Edit

Table

Edit

Table Edit :

### 1. Prompt Table & Edit:

Prompt Table is a table which consists of some valid values.

Table Edit - checks

provided

Prompts the values from prompt table and verifies the value entered in that field exists in the prompt table.

If the value provided does not exist in the prompt table, sys will throw an error msg.

### 2. Prompt Table with No Edit:

This prompts the values from prompt table but does not verify the value provided in the field exists in the prompt table.

Eid	Name	country
101	xxx	IND
102	yyy	USA
		xyz

Eid	<input type="text"/>
Name	<input type="text"/>
Country	<input type="text"/> xyz 

country - Tab

Prompt table

Magnifying symbol

This is used for fields which stores max. 2 values (or) boolean values like Yes/No true/false, 1/0 etc.

#### 4. Translate Table Edit:

This is used for fields which has translate values.

##### Table edit

Type :

Prompt Table :

Set Control Field :



Date field has diff. edit properties.

Field Name: DOJ

Required

Prompt table edit

Reasonable Date

Prompt Table :

→ This verifies the date value specified in date fields, dateline fields are within the range of 30 days above or below the current date. If the value entered is false out of this 30 days range sys will throw an warning message.

## Pages : (or) Panels :

- collection of records.
- This is a user interface to interact with people soft database.

### Types of pages :

3 types

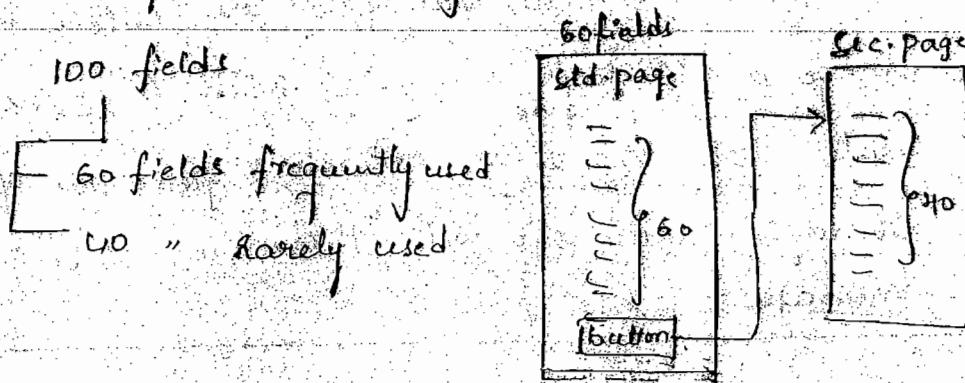
- ① Standard page
- ② Secondary page
- ③ Sub page

### Standard Page :

- Default page type.
- Can insert into component.
- can provide security.
- These are normal pages.

### Secondary Page

- cannot insert into component
- Cannot provide security.



→ U.  
ef  
Sub  
→ L.  
→ TF  
→ R

Page

Std

→ The  
rem  
unc

1. std

2. std

3. Hor

but

4. Tat

5. F

6. Gr

effective user interface.

### Sub Page :

- cannot insert into component.
- cannot provide security.
- This is used to place fields from the sub page.
- Reusability & easy maintenance.

### Page Controls (or) Page Fields :

#### Static controls

→ These controls remains static/unchanged.

1. Static Text

2. Static Image

3. Horizontal rule

4. Tab separator

5. Frame

6. Group Box

#### Data controls

→ These controls used to interact with data from database.

1. Text Box / edit box

2. Long edit box

3. Check box

4. Radio Button

5. Drop Down List / List Box

6. Push button

7. HTML Area

8. Tree

9. Image

#### Logical / processing controls

→ These controls used to maintain processing logic or pages.

1. Scroll Area

2. Grid Area

3. Scroll bar

4. Sub page

5. Secondary page

6. Active X control

7. Chart

## Static Text :

This is mainly used for page headings and to place information text on pages.

## Static Image :

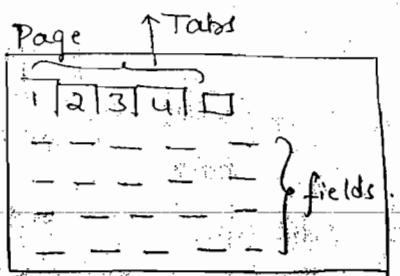
This is used to place company logos on pages.

## Horizontal rule :

Used to place horizontal line on pages.

## Tab separator :

Used to design effective user interface when fields are placed horizontally.



## Frame :

Not used from 8.0 onwards.

## Group Box :

→ This control acts as static & processing control.

→ Used to group similar type kinds of details.

→ To implement radio buttons.      ↓  
in this case it

In this case it acts as logical      acts as static  
control.      control.

Text Box / Edit Box:

→ Used for most of field types.

Long Edit Box:

→ Used for long characters.

Check Box:

→ Used for Yes/No table edit fields.

Radio Button:

→ Used for translate fields which has less no. of translate values.

Drop Down List / List Box:

→ Used for translate fields.

Push Button:

→ Used to implement logic on pages.

HTML Area:

→ we can write HTML coding on pages.

→ Used for long character fields.

Tree:

→ Used To represent data in a Hierarchy.

Image:

→ Used for image field types.

Eg: To insert employee photo, we use this image field.

13/11/08

## Logical Controls:

### Scroll Area:

Used to maintain parent child relationship on page

### Grid Area:

Used to " " " " " "

### Scrollbar:

Not used - from 8.0 onwards

### Subpage:

Used to place sub page

### Secondary page:

Used to place sec: page

### Active x controls:

Not used from 8.0 onwards

### Chart:

Used to represent data in Graphical format.

### Properties:

- 1) Page properties
- 2) Page field / page control properties

To Access page properties

click on properties icon

(or)

Alt F

To

File

It

Pag

[G]

L

④

3

4

5

6

7

8

9

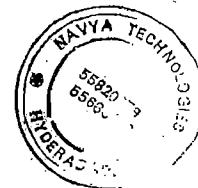
0

.

Double click on page control.

File → New  
    ↳ page

Insert a record.



Page Properties :

General Tab

Language :

Description :

short descriptions of page

Comments :

- Used for long Descriptions
- used for company commenting standards

Owner Id : used to restrict to a specific module.

Last updated

Date :

Time :

OK

cancel

Use Tab

Page-Type : → Specifies the type of page.

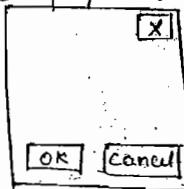
Standard page

▼

primary record

OK & cancel buttons     close Box

These gets highlighted for secondary page type.



Page-Size :

Auto-size

▼

Page sizes provided  
by people soft sys.

Width

Height

when we select custom-size, we can specify  
the width & height of the page.

when we select Auto-size, sys will take  
care about the page size.

Style :

Page style sheet

use Default

Page Background

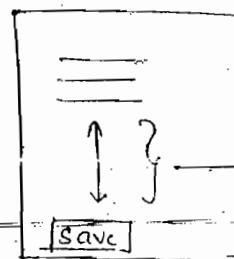
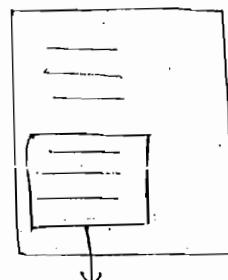
style (99%)

foreground colors etc.

3

## I Layout

### Adjust Layout for Hidden fields



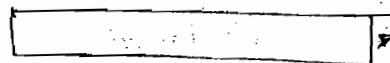
Front-end

These are hidden fields

If we select the abc  
options it adjusts the  
layout.

→ System adjusts the borders of the page till  
fields are visible, sys. removes unnecessary  
white spaces on the pages.

Popup menu:



→ we don't use this for  
page properties

## I Allow Deferred processing

Deferred Processing:

The process of bypassing system edits as well as people code written under Field Edit & field change till the user clicks on save but (or) makes a database trip.

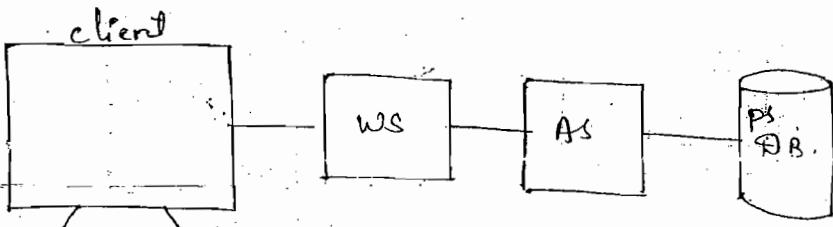
## System Edit :

Required.

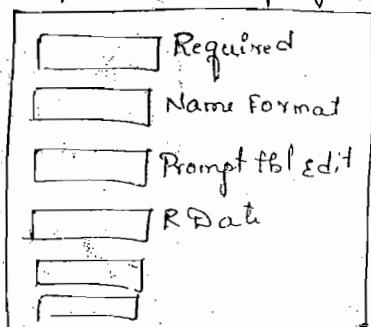
Field Format.

/ prompt table edit

Reasonable Data.



We opened a page on client



Suppose if we give Name as  
Ravi & it sends data to ws to  
AS & then it creates a error  
message & saves this in PSDB.  
Then from PSDB to AS to ws &

finally error msg is displaying.

Like this for every field it is performing the  
same process which increases the db trips.

So if we use deferred processing until we  
click on save button it will not make any  
trip. Thus we can reduce Network traffic.

→ opp. Deferred processing is interactive  
processing.

Insert → static Text



Displays  on page



Double click on this.

This opens the field properties.

Label Tab

Label Text

Text       Message catalog

(R) Message catalog:

This is a location where all messages are stored.

Message set:

collection of messages.

To identify one message set from other msg set

Every msg set has a msg set number.

To identify one message from other message in

a message set, every message has a msg no.

Message set Number: 1 to 20,000 are delivered messages.

If we want new messages we should create message set no. and we can add those message to it.

But we should not add these messages To delivered messages.

Ex.

message set no. > 20,000 → user Defined msg.

To

Navigation:

TF



(1)

People Tools → utilities → Administration → message catalog.

(2)

Eq: prompt table not defined (A7,6)  
↓      ↓  
msg set No.      msg No.

To

Field already exists on the record (A7,2).

People soft Front end



message catalog.



Add a New value



Message Set Number: 20,095

Description: Batch 25 message set

Short Description: Batch 25

message NO: 1

Severity:

message Text:

14/11/08

→

(R)

Q

②

Message

click on

↓  
This increases msg. No by 1.



② Department Details - msg.Text

③ Employee & Department Details - msg.Text

To get these messages on static text

Field properties (Static Text properties)

Message catalog

Message Set No:

Message No:

Text       Explanation

Style:

Alignment

Left

Right

Centred

} Text is placed in the box  
as per the alignment selected

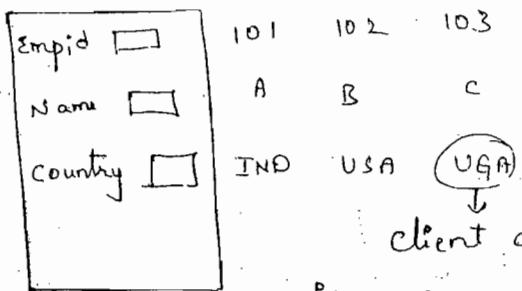
14/1/08

→ In edit box, check box, Radio buttons etc we have  
 the foll. properties

Display control field

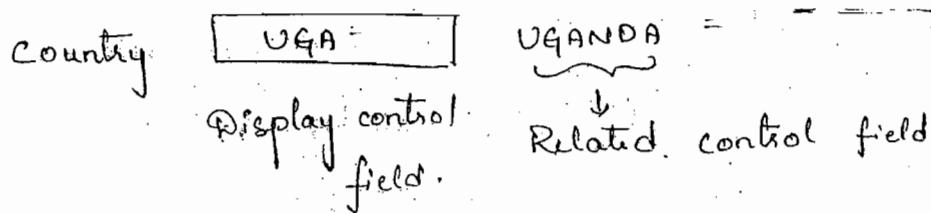
Related field

Related control field



client doesn't understand what  
is UGA.

For such type of fields we have to Display  
Uganda beside country field.



RCF is related on DCF.

whenever we change DCF value, RCF will change.  
so that field is called Related control field.

① Create a table "B25-DEF-RCF-TBL"

Emplid - K, SK

Name - ASK

Country - Prompt Table Edit - Country - Tbl

② Create a page "B25-DEF-RCF-PG"

- (5) Q
- (6) R
- steps

④ I

② Q

③ Q

④ S

⑤ Q

Dyn

R

G

G

G

G

G

G

Search Record : B25\_DCF\_RCF\_TBL

① Register the component:

Steps to be followed for the creation of DCF & RCF:

Table Name : Country-Tbl

Field Name : DESCR

② Insert Edit box

Insert :

↳ Edit box

↳ Select it & place it on page.

③ Specify Record Name : Country-Tbl

Field Name : DESCR.

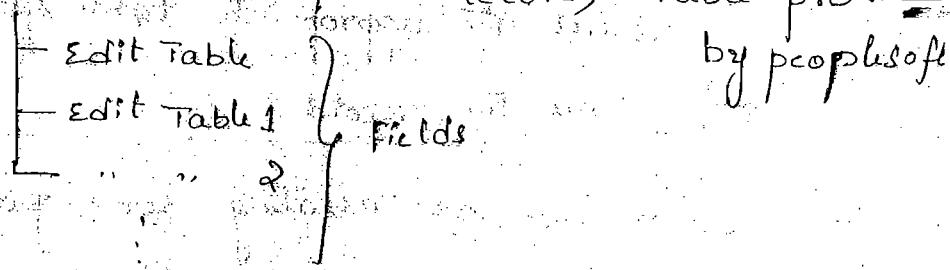
④ Select "country" field as DCF.

⑤ Select "DESCR" field as RCF.

⑥ Select "None" for label properties.

Dynamic Prompts :

DERIVED (Derived/ work Record) - Table provider



- 60-70% oracle — Editors  
PL/SQL (or) TOAD — F9
- 20% microsoft  
SQL server — Query Analyzer — F5
- 10% — Rest of db.

TOAD is more user friendly than Query Analyzer.

In oracle db, we have to specify table space

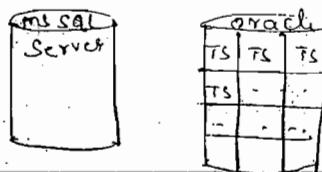
before saving the record — place where we place table

In ms-SQL server table space is 1.

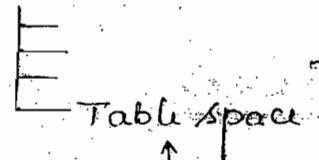
In oracle table space ~~is~~ is classified into many.

so we have to save the record in specified table space.

To specify Table space



Tools → Data Administration



Select the appropriate table space from list.

& then save the record.

→ whenever we are creating some page we need to follow some constraints.

- " 1 - Empl-Tbl
- " 2 - AFpn-Tbl
- " 3 - EDO-TBL
- " 4 - Marks-Tbl

① PeopleSoft supports max. 3 levels of parent child relationship on a page.

② Zero level record will be directly placed on a std. page & the other levels are placed using a Scroll Area or Grid Area.

③ At a specific level on a page we can have only one base record & others can be related control field table & Derived/Work record tables.

17/1/08

### Static Image:

To select static image on page

Insert → Select Static Image.

To access static image properties

↳ Double click on it.

Label

Tab

Alt tag for image

Label Type :

A10

Image

B C C

Image ID :

Fa

Dis

[ ] [ ] [ ] [ ] [ ] [ ]

Reset to Actual image size



This resets the static image box to the size of  
the image we selected.

Edit Box:

To insert it

Insert → select Edit box.

Dummy Name   
Label  Data Region

Double click on Edit box.

**Record Tab**

Field

Record Name :  B25-EmpSal-Tbl

Field Name :  EmpId.

Style :

Specifies ↓  
Border colors, Background colors etc.

Size :

Average     custom → we can set the size  
of edit box

B Play

- Auto
- Left
- Right

The data (LHS & RHS)

→ characters: Left → Right

→ numbers: Right → Left

Family Name:  Display Name

### Display options

Display zero → used for no type fields to place zero if there is no data.

password → used for hiding data. Shows data as \*.

Show prompt → shows a magnifying glass button. we get data from prompt.

Auto Fill.

These 2 comes with  
Fill character  combinations If no data present entire field is filled with symbol given

Display Time zone - Displays a dropdown list indicated time zone specifies the time of particular zone

Display century - Data time fields to specify the century on RHS

Currency symbol

Salary

100

INR

USD

1000 separator → used for number fields

20,005,000.00

System automatically place commas after every  
3 digits from decimal pt towards LHS

Auto Decimal → used for no. fields

Salary

must be integers & decimals

If 5000.00 is given it stored as  
5000.00

If 5000.1 (that is there we have decimal  
point) in this case it appends zero

7.3

5000.100

Enable Spell check → Sys checks spellings  
with dictionary.

Label Tab

↳ specifies about label part

Type:

None  Text  RFT short  RFT long

used to display short labels used to display long labels.

Text :



Label ID :



Specifies the label to be displayed on front end

Style :

Alignment position

left  left

center  TOP

Right  custom

Places the label in the req. position w.r.t  
To Edit box.

Emplid

} Displays like this  
on page.

#### Display options

First occurs only

No colon

Use Tab

Display only — we make the field as  
uneditable

invisible — Both label & edit box are  
hidden

show label

modifiable by  
java script

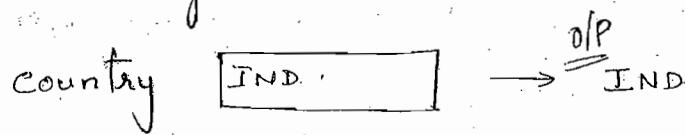
multi currency - To Display currency sym. on LHS  
Field

Display control Field

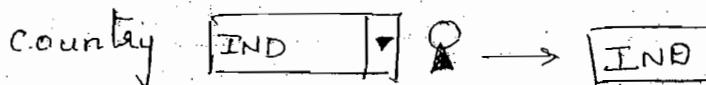
Related Field

Display only Appearance — This should be done

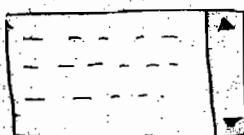
Text only when display only option is selected.



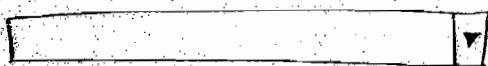
Disabled Edit control



wrap long words



Related control Field :



~~Eq:~~

Insert

↳ Fields

↳ emplid, → K, searchkey

Name → Alt " "

Business-unit

with Edi

Save BAS-DCF-RCF-TBL.

Create a page.

↳ Drag & Drop all fields.

Save → BAS-DCF-RCF-PG.

Create a comp.

↳ Drag & Drop page

Save → BAS-DCF-RCF-CMPY.

Search Record: BAS-DCF-RCF-TBL

Register the component.

If we select prompt table with edit, if we enter any data that is not present in prompt table it will <sup>not</sup> allow us to enter. It validates the data that is present in prompt table.

DCF & RCF For country:

Country  Add a edit box.

Edit box prop.

RecordName: country -Tb

Field : Descr.

} Go to Label Tab

Type:

None

Double click on country field.

↳ mark it as RCF.

Double click on edit box.

↳ use Tab

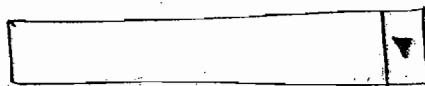
↳ mark it as Related Field

Cond

when we select Related Field it automatically

Selects  Displays only &

Related control field.



Select \* from PS-BUS-UNIT-TBL-FS

RCF & RCF For Bu Field

Business unit field.

↳ prop.

↳ prompt table

↳ BUS-UNIT-TBL-FS

Insert edit box beside BU

RCF : BUS-UNIT-TBL-FS

: Descr.

Label Tab

↳ Type @ None

go

19/11/08

P

o

o

o

o

E

To

u

u

u

↳ mark as DCF

click on edit box

↳ mark as RCF



Conditions to be followed while creating DCF & RCF

- 1) Display control field option must be selected prior to the related control field option.
- 2) The order of DCF must be prior to the order of RCF.

Order Tab on page specifies where it should go after clicking on Tab.

19/11/08

Pop up Menu → This will be explained clearly in Menu Concept:

Emplid



c1

[c2]

Goes to searchpa  
of this comp

c3

used mainly to move from one component to other component

Allow Deferred processing

Set component changed

Used for programming purpose

If this is selected, in the backend it maintains the flags to verify whether the data in that particular field has been changed or not.

General Tab.

Eid	101
Name	A
Basic	100
TA	200
DA	300
HRA	400
Income Tax	50
Total	995

If Basic field value is changed then Total field value is also changed.  
i.e one field value is dependent on other field.

If we have 1000 fields on page its diff. to find which

field value is changed.

For this we have

Page Field Name

Enable as page Anchor → it places an anchor symbol beside the field.

This is used to move from one field to other field on the same page.

Insert → check box

→ Drop it onto page.

Data region  Dummy Name Label

To Access it Double click on it



### Record Tab

Field

Record Name : B24-EmpSal-Tbl

Field Name : empId

Field Value

on value :

Y, Yes, T, TRUE	1	Y
-----------------	---	---

off value : N, NO, F, FALSE, 0

### Label Tab

Type :

None

Text

RTF

Short

Label Text :

Text :

Label ID :

Style :

Location :

Left

Right

Alignment

Left

Centred

Right

### Use Tab

- Display only
- multi currency field
- invisible
- DCF
- show label
- RF
- modified by  
Java script

### RCF

### Popup menu

Allow. Deferred processing

Set comp. changed

### General Tab

#### Page Field Name

Enable as page anchor

### List Box

Insert → List Box

↳ Drop it on page

Properties → Double click on it.

### Record Tab

Lat short

Lat long

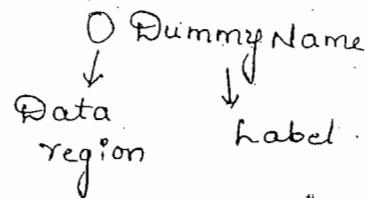
prompt table field

Displayed Text

field

Insert → Radio Button

↳ Drop it on page.



**Record Tab.**

Field : MAR-STATUS → This has translate

Record : B&B-Empl-Tbl values

value :

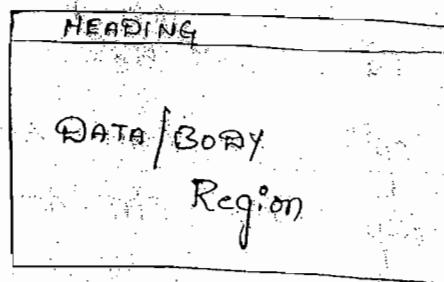
Select the value of field Name specific

→ Radio Buttons are used for translate fields which has less no. of translate values.

**Group Box**

Insert → Group Box

↳ Drop it on page & set the size of it.



Properties → Double click on it.

If we have given Record Name & field Name of

Record Tab it prints the field Name what we selected, on the heading region of group box.

Label Tab

Print

Body style → Sets the background color, border colors etc of group box.

Hide Borders

Adjust layout for hidden fields

Print

Use Tab

Hide all fields when Group Box Hidden

C

(ii)  Collapsible Data Area

Eg: If we have 10 group boxes, if we want to see heading of 1<sup>st</sup> & last group box, the client has to move the scroll bar.

If we select this option, we can see only the headings of all group boxes with the arrow button. If we click on arrow button it will display the data.

The particular group box will be in expanded state, all other " " are displaying only headings.

### Push Button

Insert → push / Hyperlink



Properties → Double click on it.

Type Tab

push Button     Hyperlink

Search    Advanced search  
↓              ↓  
Push Button    Hyperlink

Only appearance is changed

Destination:

①

External link → Access sites which are outside

static.    of people soft system

Dynamic

URL ID :

Destination: Internal Link → used to access  
②              a page which is within ps sys.

Internal Link

(3)

Portal

Use current

(4)

Eg: we are registering with employee.

(5)

Node:

LOCAL-NODE

Menu:

B23-Test-Menu

Comp:

B23-Test-Comp

Market: FBL

Page

B23-Actual-Pg.

So

Action:

→ specifies to what purpose we are opening the page

(1)

Add

update / display

Si

Se

S

Si

Eg:

C1

Dept Pg

C2

Personal Details

  
Name:  
Add:  

(6)

(7)

(8)

use data from current page in search.

If we select this, it picks the common field

Value (from Dept Pg) & displays the same  
field value in other page (i.e. in personal details)

④ people code command — used to execute people code.

⑤ process:

process

Type : SQRReport

Name : PAY14.

If push button is clicked then this (PAY14) SQR report prgm is executed.

Process Type	App Engine	Database Agent
SQR Process	COBOL SQL	Message Agent API
SQR Report	CRY Online	nVISION - Report
SQR Report FOR WF Delivery	Crystal	nVISION - Report Book
WINWORD	Cube Builder	optimization engine
	Data Mover	PSJOB

⑥ prompt Action:

Action

Action Type : prompt

⑦ scroll Action → Action Type — Bottom, prev.

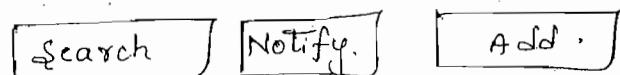
⑧ secondary page

when we click on push button we have to go to sec page.

Sec. Page Page

## ⑨ Tool box Actions

21/11/

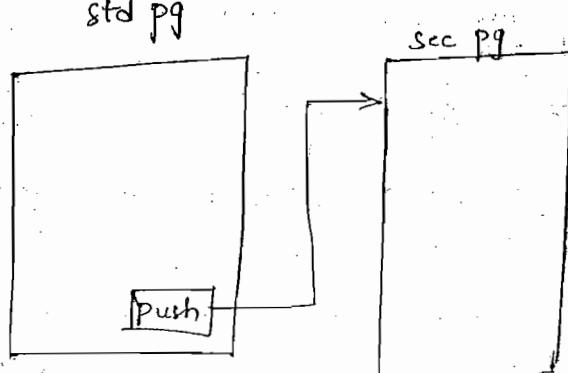


## ⑩ Instant Messaging Action

Action Type : Instant messaging.

Used to send msg from one sys. to other.

Enable when page is Display only



If in the std pg, we selected all the fields

along with push button as Display only it will  
not navigate to sec pg.

so even if all the fields on the std pg are  
Display only. If we selected this option, the  
push button has its functionality to navigate to  
other pg.

Open in New window

### Scroll Area

- Used to maintain parent child relationship
- Used for the levels of records which are not equal to zero & last level on page.

→ we can design our own layout



0 - Dept ID TBL

1 - Emp ID TBL

2 - Dependent ID TBL

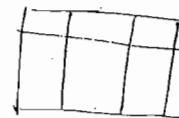
3 - Education TBL

4 - Marks TBL

### Grid Area

- Used to maintain parent child relations
- Used for the last level record & level not equal to zero.

→ we can place field as column in grid as we can't design our own layout



0 level - standard page

1 level - scroll Area

2 level - scroll Area

3 level - grid Area.

0 level - std page ; last level - grid Area

Middle - scroll Area

If last level of record consists of subpage then we use scroll Area.

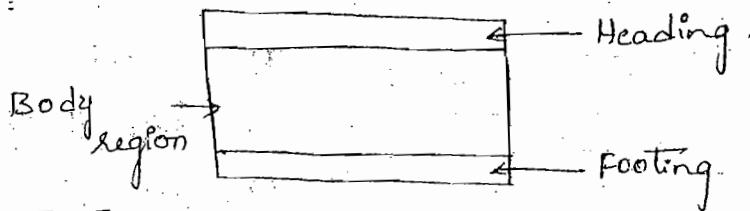
→ we can't export to the excel sheet

→ we can export data to excel sheet

→ we can place subpage → we can't place subpage  
if the last level of record  
consist of sub records,   
instead of grid area  
we use scroll area.

### Scroll Area:

Insert → scroll Area → click on page properties.



General Tab

I) Page field Name → move one field to another field in same page

Occurs level → use to specify the level of record fields that are being placed in the scroll area

Occurs count → use to specify the no. of rows to be viewed default

Eg: Message catalog count = 1

unlimited occurs count (rows) → bypass the occurs count property & displays all the list of rows from scroll Area

Display Header

Header Area

Display title

Display Navigation

Body Area

Display Row Action Buttons

Background style

Show Row Separator  Show Border

Adjust layout for hidden field

Display footer

Footer Area

Display Row Action Button

User Tab

Data options

No Auto select → can't move one row to another

No Auto update

No Row Insert →  button not work.

No Row Delete →  button not work.

Allow Deferred processing

Allow Multirow Insert

## Row Selection Indicator

2H11e

C

→ C

→ U

P

G

F

① No Selection Indicator

② multi row

③ single row

## Component :

- collection of pages.
- used to move pages to portal window.

## Properties :

### General Tab

Description : Short Description

Comments : \_\_\_\_\_

Long Description

company commenting stds.

Owner Id : used to restrict to specific module.

Last Updated

Date / Time

User

### Use Tab

Search Record :

- used for searching process.
- used to provide row level security.

Add Search Record :

- used for adding new set of data

If Add & search record is not given then it takes same as search record.

Find existing value — From search record

(Search Key &, Alt. Search Key)

On!

Add a New value — From add search record

(Search key)

Force search processing



Used to restrict the company to go under searching process.

③ R

→ US

Component Buffer is temp. memory location for people soft system.

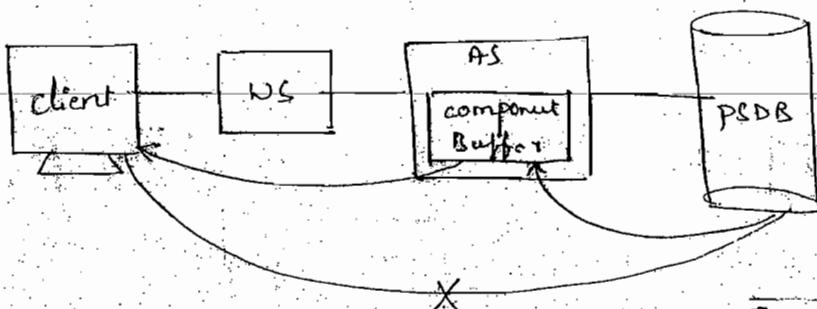
da

F

→

a

hi



3-Tier Execution Location

comp. build

○ client

○ APP' server

① Default

comp. save

○ client

○ APP' server

○ APP' server (with edits)

① Default (APP' server)

Effect

→ H

If we select this, comp. will become Displa  
only.

Include in Navigation

↓  
with this only we can register the comp.

Mandatory Spell check.

used to perform spell checking on comp.

## ② R Effective Dates:

→ Used to maintain data  
datewise (history, current,  
future)

Emplid	Name	Salary
101	A	1000
102	B	2000
103	C	3000
104	D	4000

New → 104      After 6m  
                  "        5000  
                  "        100

→ This is maintained using  
a field 'EFFDT' as one of the  
high level key field in Table.

Emplid	EFFDT	Name	Salary
101	01/01/08	A	1000
102	01/06/08	B	2000
103	01/06/08	C	3000
104	01/01/08	D	4000
102	01/02/08	B	5000
104	01/06/08	D	8000
104	01/01/09	D	10000

## Effective Date Table:

→ A table consisting of Effective date as high

level key field known as Eff. Dated Table.

This is used to maintain data in current, history & future rows.

### Current Effective Dated Row:

The row of data which is effective as of today is called as current eff. dated row.

### History Rows:

The rows of data where effective date value is less than effective date value of current effective dated row are called as History rows.

### Future Rows:

The rows of data where effective date value is greater than eff. date value of current effective dated row are called as Future rows.

### Conditions to identify Current effective dated row:

- 1) Identify the list of rows where effective date value is less than current date.
- 2) Out of the rows selected in step 1, pick the highest eff. date value.

### Effective Dated Query:

The select stmt used to identify current effective dated rows in an effective dated table

EmplId	EffDT	Name	Salary
101	01/01/08	A	100
101	01/06/08	A	200
101	01/01/09	A	300
102	01/01/08	B	250
102	01/08/08	B	250
102	01/09/08	B	450
102	01/12/08	R	550
103	01/01/08	C	110
103	01/06/08	C	210
103	01/09/08	C	310

Eq:

Select \* from PS\_Empl-Tbl A } Main Query  
 where A.EFFDT =

{ Select max(B.EFFDT) From PS\_Empl-Tbl B

where B.EmplId = A.EmplId

AND B.EFFDT <= sysdate )

Subquery

Select \* from PS-JOB A

where A.EFFDT =

{ Select max(B.EFFDT) From PS-JOB B

where B.EmplId = A.EmplId

AND B.EFFDT <= GETDATE()



25/11/08

## Effective Date & Effective Sequence:

→ Used to store multiple transactions happened on the same day.

Emplid	EffDt	EffSeq	Name	Salary
101	01/01/08	1	A	100
101	01/06/08	1	A	200
101	01/10/08	1	A	300
101	01/10/08	2	A	400
101	01/10/08	3	A	500
101	01/01/09	1	A	600
102	01/10/08	1	B	150
102	01/10/08	2	B	250
102	01/11/08	2	B	350
102	01/11/08	1	B	450

## Effective Sequence Query:

Select \* from PS-Empl-Tbl A

where A.EFFDT =

(Select Max (B.EFFDT) From ps-Empl-Tbl B

where B.emplid = A.emplid

AND B.EFFDT <= sysdate)

AND A.EFFSEQ =

(Select Max (C.EFFSEQ) From ps-Empl-Tbl C

where B.emplid = A.emplid

AND B.EFFDT = A.EFFDT)

(II)

where A.EFFDT =

(Select Max(B.EFFDT) From PS-JOB B

where B.EmplId = A.EmplId

AND B.Empl-Rcd = A.Empl-Rcd

AND B.EFFDT <= sysdate)

} we have to m  
all the key fi  
above effdt.  
field.

AND A.EFFSEQ =

(Select Max(C.EFFSEQ) From PS-JOB C

where C.EmplId = A.EmplId

AND C.Empl-Rcd = A.Empl-Rcd

AND C.EFFDT = A.EFFDT)

Actions

[ components  Tab ]

Add

update / display

update / display

↓

All

Adds only current  
row

correction

Actions ↓	rows →	History	Current	Future
Add		Cannot see / Add/update rows	Add	cannot see / Add/update
Update / display		cannot see / Add/update	Add/see / update	Add/see / update
update / display All		see / can Add / update	Add/see / update	Add/see / update
Correction		see / Add / update	Add / see / update	Add / see / update

Eg: Recruiter — Add

To

Manager — update/Display

Sr. Manager — update/Display All

Director — correction.

### [Internet] Tab

#### Search page

To

primary Action  New  Search

D

Default search Action :  update/display

F

#### Default search/lookup type

WORKING  
FROM  
SYNCH

Basic  Advanced

E

Allow Action mode selection → we can move from one mode to other.

3 days

#### Multi-page Navigation

Co

Display folder tabs [top]

W

Eg: Page 1 Page 2

as

Display hyperlinks [bottom]

at

Eg: Page 1 Page 2

the

#### Processing mode

in

Interactive  Deferred

the

Allow Expert entry

the

↳ Buttons which we see on the bottom of the page.

Notify

↳ used to send a email from our people soft sys to other sys.

Smtp should be activated on other sys

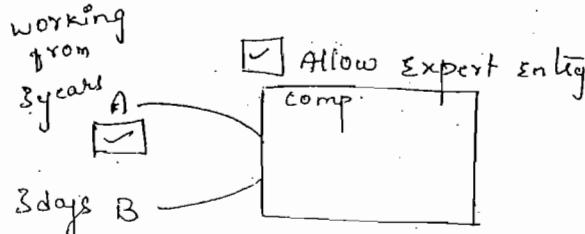
### Page bar

Help Link

New window Link.

Copy URL Link

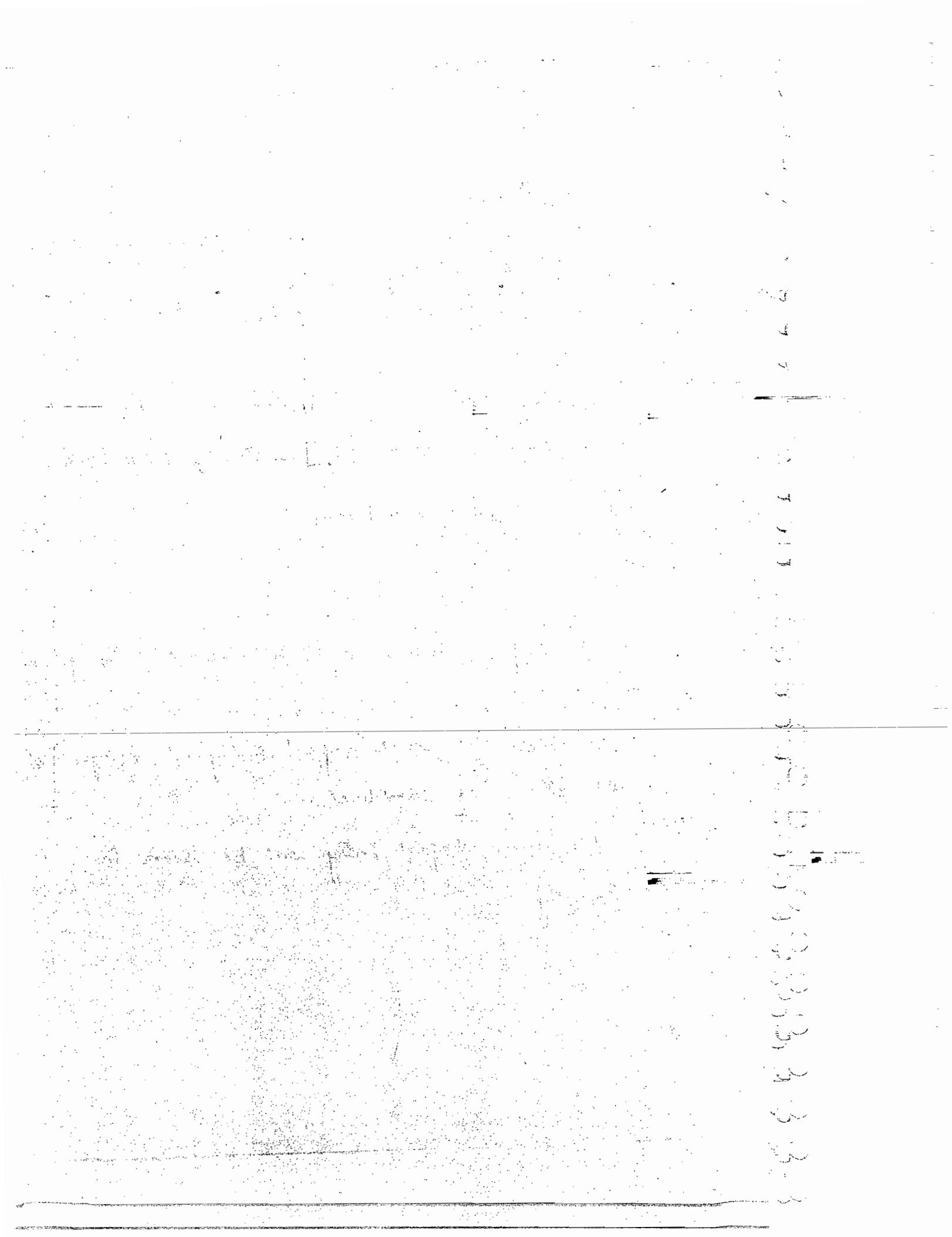
Customize page link.



sys. verifies whether he is a expert entry person (or) not.

we need to select Expert entry at comp. level as well as at User level.

User level Expert entry will be seen in Security.



26/11/08

## Application Designer

### Menu:

- collection of components.
- Used to register a component.
- Used to navigate (or) move from one comp. to other Component

### Types:

- ① Standard Menu
- ② Popup Menu

#### Standard Menu: [Default menu]

- This is in horizontal fashion.
- Used to register a component.

#### Popup Menu:

- This is in vertical fashion.
- Used to navigate from one comp. to other.

### To create a Menu

File → New → Select Menu



Click on OK.



Select the type of menu

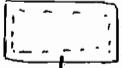
① Std

② Popup menu

OK



→ This opens a window

File Edit ...  Lang Help

Bar Name

Bar

Process → used to place/run control components

USE —

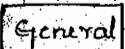
To add Bar item, click on Space & add the req. bar item.

Drag & Drop the req. component to the bar item.

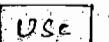
Properties:

- ① Menu Definition properties [Alt + ↓ or click prop. icon]
- ② Bar Item properties [Double click on Bar Item]
- ③ Menu Item properties [ " " " component ]

Menu Def. properties :

 General Tab

— Same as Before

 Use Tab

These properties are not req.

Before 8.4 Version in the portal window it displays menu names instead of folders so for that purpose this Tab values are used.

Menu

Me

Tr  
G

CPE

CE

CE

CE

CE

CE

## Bar Item Properties :

Name : process

Label : process

OK      cancel

## Menu Item properties :

### MenuItem :

Name : B25-Test-cmpt

Label : Test comp. Reg.

### Type :

O component

O pcode — mainly used for popup menu.  
— Rarely used.

O separator — used to separate one set of  
cmpts from other set of cmpts. places a  
horizontal line.

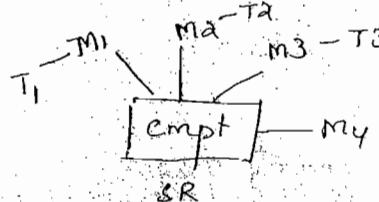
### Component :

Name : B25-Test-cmpt

Market : GBL

Search Record : B24-EmptySal-T61

override : we can override the search  
record.



Mainly use to change the search record for diff.

→ If  
just

Menu's.

### Folder:

#### Creation:

##### 1) Navigation

- PeopleTools → portal → structure & content.

##### 2) To Add New Folder

###### Add Folder



This opens a separate page.

##### 3) Name : Batch 25

Label : Batch 25 — This will be the folder Name

Long Descriptions → when we place mouse on to folder

it displays some desc.

This is test folder

product :

Seq. Number : 9999

↑ At which position we want to see our  
folder

##### A) Folder security page

public

→ If we want to change the position of our folder just click on edit & give the value.

### Component Registration:

Two ways.

- ① click on reg. icon & follow the steps
- ② Go to menu → Select comp.

↓  
Rt. click.

↓  
Register Menu Item.

If we want to register the component to our own folder then in the wizard select the folder name.

Content Ref. Label : Department Details

↑  
This can be anything.

Long Description : Department Details using subrecords

↑  
This will be displayed when we placed cursor on to the label.

Seq. Number : This can be any number.

Product : PT

Always use default local node

PT-local

Next

In the Next window,

Add project

Menu

Registry entry

Permission list

**Finish**



3.)

After doing this, it adds permission lists and portal registry structures folder to the upgrade tab in the App' designer of project name.

27/11/08

PROJECT:

[V1-DEPT-EMPL-PROJECT]

4.)

1) XX-DEPT-TBL (SQL Table)

XX-DEPTID - char(2), K, SK

XX-DNAME - char(40), ASK

XX-LOC - char(3)

XX-MGRID - char(3) Reg. [XX-Emplid]

2) XX-EMPL-TBL (SQL Table)

XX-DEPTID - K, SK

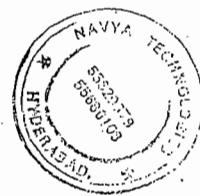
XX-EMPLID - char(3), K, SK

XX-ENAME - char(40), Name, ASK

XX-DOT - Date Reasonable Date

XX-MAR-STATUS char(3) XLAT

5.)



XX-BASIC — Num (7.2)  
XX-TA — Num (7.2) 800 - constant  
XX-DA — Num (7.2)  
XX-HRA — Num (7.2)  
XX-DOD — Date  
XX-DOB — Date

3) XX-DEPEN-TBL (SQL Table)

XX-DEPTID — K, SK  
XX-EMPLID — K, SK  
XX-DEPENID — char(4), K, SK  
XX-NAME — ASR [XX-ENAME]  
XX-AGE — Nbr(3)  
XX-RELATION — char(3), XLAT  
XX-ADDR-SBR — subrecord

4) XX-ADDR-SBR (sub record)

XX-Street1 — char(40) [XX-<sup>NAME</sup>-DPT]  
XX-Street2 — char(40)

XX-city — char(3) [XX-LOC]

XX-state — char(2)

Country — char(3) — use existing field.

5) XX-FDU-TBL (SQL Table) Edit → prompt table

XX-DEPTID — K, SK  
XX-EMPLID — K, SK  
XX-DEPENID — K, SK  
XX-CLASSID — char(40) K, SK (SSC, BTech)  
XX-INST\_NAME — char(40) ASR [XX-DEPTNAME]

XX-MARKS — NBY(3) [XX-AGE] Diff. labels

XX-GRADE — CHAR(1) XLAT

### 6) XX-DEPT-SRCH-V [SQL VIEW]

XX-DEPTID — K, SK

XX-DNAME — ASK

#### SQL

Select XX-DeptId, XX-DName, From

PS-XX-DEPT-TBL where XX-DeptId <> '10'.

### XX-DEPT-EMPL-PG [std. pg.]

static Text → Employee & Dept Details

DeptId	[ ]	Deptname	[ ]
Loc	[ ]	*Mgrid	[ ]

#### Level 1 Employee Details — Scroll Area

Emplid	[ ]	Name	[ ]	DOJ	[ ]	
Mar_Status	Basic	[ ]	TA	[ ]	DA	[ ]
HRA	[ ]	TOTAL	[ ]	DWR	[ ]	
				Push	[ ]	

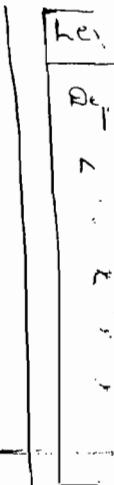
Secondary page

DWR  
Derived  
XX-TOTAL-DWK (soft table)

XX-TOTAL-NBY (8:2)

XX-EMPL-SEC-PS

DOD	[ ]
DOR	[ ]



Com

P

S

1/12/08

In

write

B25

U

U

U

write

U

Level 2                      Dependent Details

Depenid	<input type="text"/>
Name	<input type="text"/>
Age	<input type="text"/>
Relation	<input type="text"/>

xx - ADDR - S.BP → Subpage

Street 1	<input type="text"/>
Street 2	<input type="text"/>
city	<input type="text"/>
stat	<input type="text"/>
country	<input type="text"/> & <input type="text"/>

DCE      RCF

Level 3                      Education Details

classid	InstitutionName	marks	Grade

Component (xx - Dept - Empl - cmpt)

Page : xx - Dept - Empl - pg

S. Record : xx - Dept - Tbl.

11/12/08

In HRA field of Empl-Tbl, Field change event

Write the code

B25 - Total - Dwk. Total. value =

Write the same code in HRA field Row - Init event

Give the search record as B25-Dept-SCHR-V. It restricts the display of 10 department on search page as the code we have written. Means it is providing row level security.

1) C

2) S

3) R

4) P

5) T

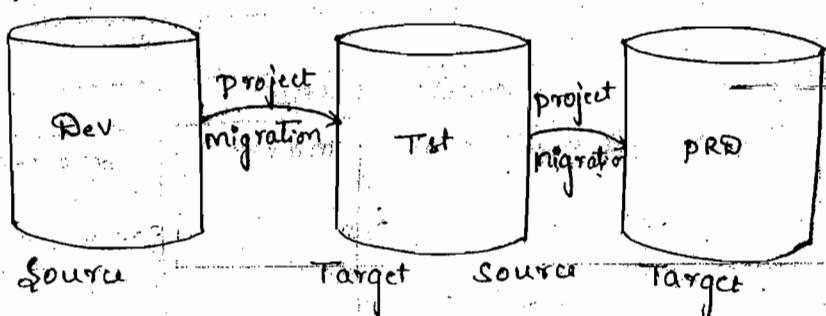
6) I

7) U

8) D

O/P:

### Migration:



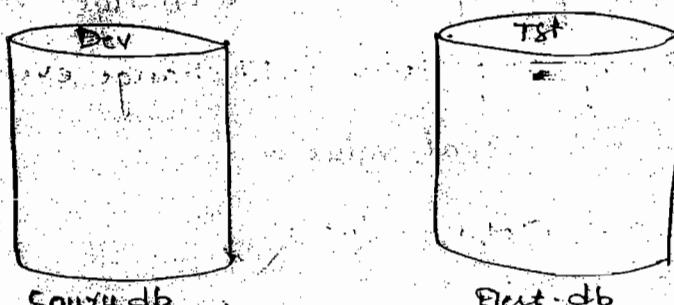
→ project is the only object we can move from one db to other.

→ moving is called project migration.

The project from one db to other db.

2/12/08

### Steps to be followed for migration:



P1

R1 (New Rec)

  |  
  F1, F2, F3

JOB

  |  
  Some fields (F1, F2, F3)

Person  |  
  |  
  F4, F5, F6 (Instead of F6 we add F8)

use

but

done

- v. It search  
it is
- 1) Create a project by inserting the necessary objects.
- 2) Run "Compare & Report".
- 3) Analyze output.
- 4) Mark appropriate objects that need to be migrated.
- 5) Project Migration & Flags verification.

obj:

ObjName	Source	Destination
R1 Record	F1 } F2 } Green F3 }	≡ }
JOB Record	Means these are not in Dest: R1	we don't find anything b'coz R1 is new Record
Person Record	Some Fields  F4 } F5 } Green F6 }	Some fields

we need to migrate R1 Record & JOB Record,

but no need to " Person record b'coz we haven't  
done any changes.

B25\_Dept\_Empl\_project → This is to be migrated

Open reg. project.

Tools → compare & Report

Database Name :

User ID :

Password :

} Give the dest. db  
names.

upgrade Tab

OK ↴

[comp, records, fields, SQL :..]

To select the objects for migration, make sure ..

Upgrade — is selected

Done — is deselected.

Not to migrate

upgrade — Deselect

Done — Select

To migrate project

Tools → copy project

↳ To Database.

Db. Name.

User ID :

Password :

} Dest. DB values

OK ↴

### upgrade Tab

Flags verification. Check upgrade & done checkboxes

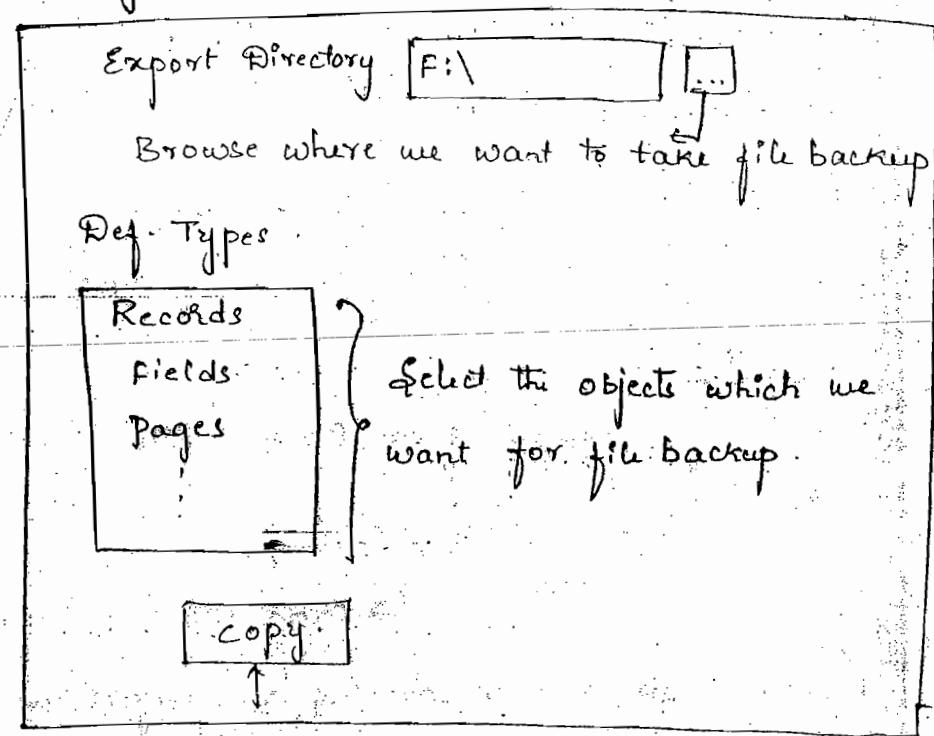
If both Upgrade & done are selected then migration is done properly.

### File Backup of project:

To take a file backup

Tools → copy project  
↳ To file

copy to file



F:\ B&S-Dept-Empl-project

(3)

This consists of 2 files

① config. file

② XML. file.

③ R

④ Backup file will be created with our projectName. →

→ To Delete fields from db.

File → New → Delete.

Def : Field.

Name : B25-AGE..

Yes

→ If we delete the fields from the project, it will not remove from db.

→ Remove the fields from the project.

Recover:

To get project from Backup file

① Tools → copy project

↳ From file

② When we are recovering the project we will recover the project from config. file.

Select the file [ B25-Dept-Empl-project ]

↳

config. file

③ Replace the existing objects



→ If we want to change any field length.

Edit → Find Ref. ref. [for that field]

make list of all the Records.

change the field length.

Alter all the Records.

Pages will not be effected.

CF " " " "

If it is Exalto ci it gets effected then

Create a New template.

Query view also need to be altered.

people code will not be effected.

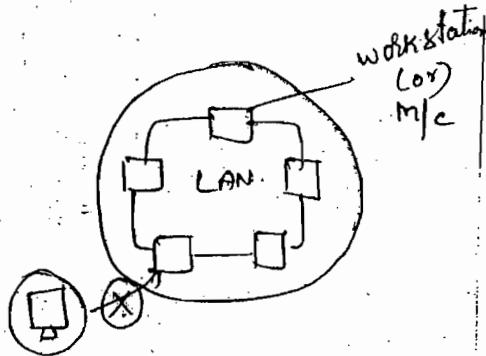
Identify the objects that need to be affected by this change — Impact Analysis.

3/12/08

## Security

Types:

- ① Network security.
- ② operating sys. security.
- ③ Database security.
- ④ App' security.



Network Security:

- Maintained using Routers (Technology Department)
- Secure the systems from the ext. sys.

operating sys. security:

- Maintained using Network IDs (Tech. Department)

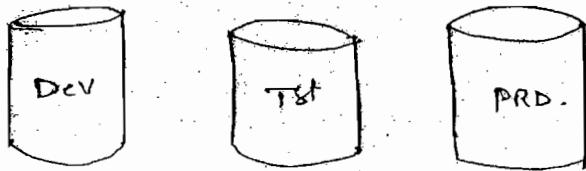
User ID / Network ID / LAN ID

Password

Domain

Database Security:

- maintained using Database user IDs (DBA)



App

a)

b)

c)

d)

App

① [

→ S

→ R

② F

→

③ M

→

Q

→

D

→

C

## Application security :

- a) Row level security .
- b) Field " " .
- c) Menu " " .
- d) Definition " " .

## Application security of people soft :

### ① Row level security :

Restricted some rows without accessing .

→ Securing few rows of data in a table is row level security .

→ This is maintained using search records .

### ② Field level security :

→ Securing few fields in a row of data is field level security .

→ we can obtain <sup>by</sup> people code .

### ③ Menu level security :

→ Securing pages & components is menu level security .

→ Maintained using security objects .

### ④ Definition level security :

→ Securing ps def's from other developers is called Def level security .

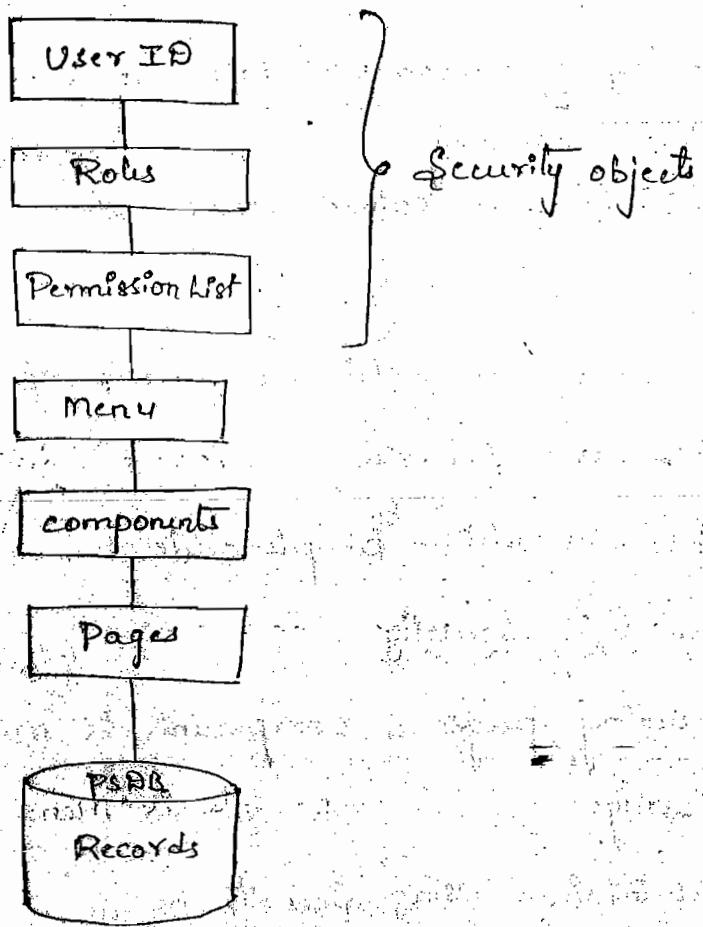
→ maintained using Def. security tool.

### Menu level security :

→ Collection of Menus that can be accessed is called permission list.

→ collection of permission list is called Roles

→ collection of Roles is called user ID



### Permission list :

Now! PT → Security → permissions & Roles

↓  
Permission List

Open AEAR1000 permission List.

**General Tab**

is called

PL : AEAR1000

Can start Application Server?

Allow password to be enabled?

Specifies whether email password is to be sent to mail or not.

① Never Time-out

② Specific

**Pages Tab**

Select menu (any) & click on edit component:

Authorized

Edit page

If comp has no. of pages then give the access to the reg. pages:

Authorized is selected then u have given permission to access

Display only → This is selected when we want the page to be read-only.

Actions

Add     update / Display     update / DisplayAll

Correction.

### people tools Tab

specify what are all the tools that a person can access.

### process Tab.

Reg. To provide security for our programs (or)

process.

### Sign-on Times Tab.

At what time a person has access to this page.

Restrict sign-on-times for pl.

### C.I Tab

This is used to provide security for C.I.

### Message Monitor Tab

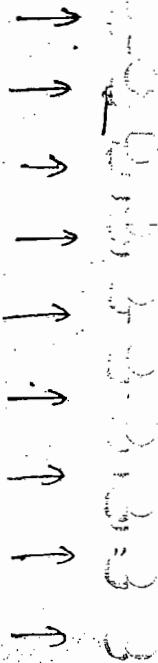
provide message security.

### Web Libraries Tab

is a place where the functionality of web page coding is present.

provide web library access

Adv



### Query Tab.

Explained during Query manager.

### mass change

we can update huge amount of data from the front end.

provide Mass change template security.

### Links Tab

Not imp.

### Audit Tab

### PL Libraries

we can see the data of pl.

### Advantages of permission lists:

- can start As
- provide page access
- " PT Access
- " process group security.
- " sign-on-time
- " security to c)
- " msg security
- " web library Access
- " mass change template security

4/12/08:

## Roles:

→ Collection of permission lists

### Nav:

People Tools → Security → Permissions & Roles

Role

Open any Role (Applicant)

General Tab.

Role Name: Applicant

Description: Internal Applicant

Permissions

consists of permission list who have access for  
this role.

Workflow

- Allow Notification — a person is a workflow
- Allow Recipient Lookup user you can send Email Notification
- Use Query to Route workflow

## User ID:

### Nav:

People Tools → Security → User profiles

User profiles

I

use

he

of

"

{



En

use

ot

Ex

Th

pe

h

u

c

u

u

In company's every person will have a diff account.

#### Logon Information

Symbolic ID: sa1

User ID Alias:  

Password:  

Confirm password:  

Account Locked out?

↓

while resetting password

make sure this is

unchecked (when our  
login page is locked)

Expire password at next login.

↓  
make sure this is checked (when our  
login page is locked)

These are in encrypted format.

Enable Expert Entry

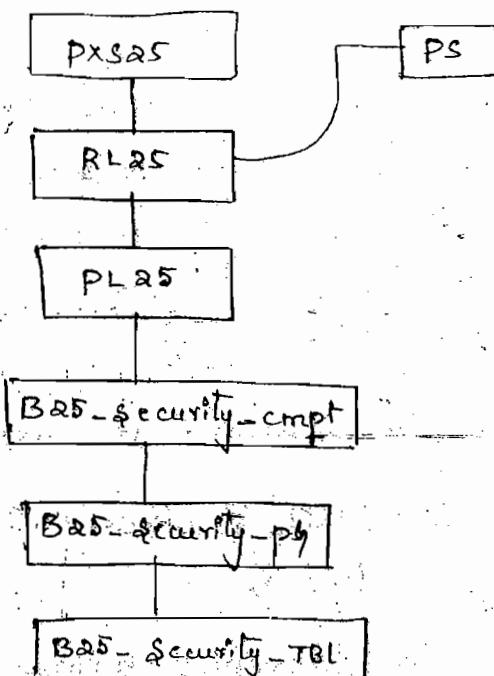
↳ along with this if we specify Expert  
Entry level at component level then that particular

user is allowed to perform bulk insertions.

Otherwise if at component level if we select  
Expert entry & here if we unchecked this

that particular user is not allowed to  
perform bulk operations.

Eq:



→ create a record, page, component

→ To create ph:

PT → Security → permissions & Roles

Rey

Defin

Gr1

Gr2

G

G

G

G

G

G

G

G

G

G

G

Add New PL → PL25

Add

Desc : Batch25 PL

→ create a New Role -

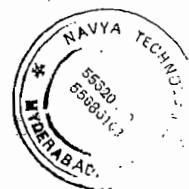
with Name - RL25

Desc :

→ Create New User ID

copy user profiles (ps)

Add New user ID : pxs25



→ open Role

↳ Go to [permission list] Tab

↳ Add [pxs25] To it

→ open user profiles

↳ Go to [Role] Tab

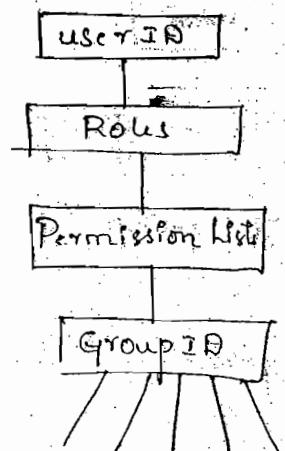
↳ Add PL25 To it

Register the component under this permission list  
(PL25)

Definition level security :

Group → collection of ps objects

GroupID is a name given for a group.



collection of ps objects

open App' designer. [To Access Def. Security].

Go

↳ Def. Security.

This open ps Def. Security window.

① Create a group

File → New group

Select the definitions.  Records

Displays all the list of records.

Select the records need to be secured.

Like wise do for pages, comp... for the def. u need.

② File → Save

↳ Save group as

Group25

Now we created a group.

③ To see the objects in group

File → Open

↳ Group25.

④ Adding group to pl.

File → Open → pl

Displays all pl. → Select the pl & [OK]

(pl25)



Select Group25



Role change

→ Display only.

To make Display only as 'No' Remove that group25 from list & again add.

(W.H.)

### PeopleSoft Query Manager:

- Proprietary Tool.
- Reporting Tool.
- This is a GUI Tool.
- This is used to design Select statements.
- " " " for reporting purpose.
- Retrieves data in rows & columns.
- Used to export output to excel sheet.
- Used in Crystal reports.
- " " " vision reports.
- " " " workflow.
- Used in 2-tier & n-tier.

5/12/08

Eg:

## Types of Select statement:

### ① Simple Select:

Used to retrieve data with a single table.

### ② Select with Distinct:

Used to get unique o/p dataset

### ③ Select with Order By:

Used to get data in asc. or desc. w.r.t some fields.

### ④ Select with prompts:

Used to provide runtime parameters to the select stat.

### ⑤ Select with Joins:

Used to get data from multiple tables.

#### a) Inner Join (or) Equi Join

#### b) Outer Join

##### i) Left Outer Join

##### ii) Right Outer Join

#### c) Cartesian product

#### d) Self Join

& get  
in c

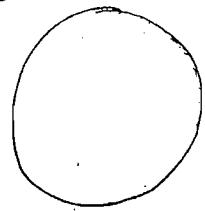
ROJ  
com

addit  
on R

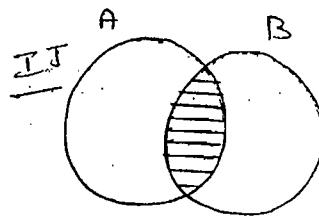
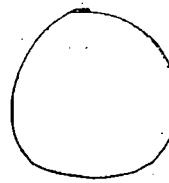
SJ  
L  
R

→ we  
Peg  
pick

Eq: A = 10 rows



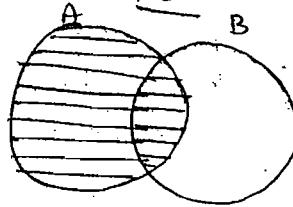
B = 5 rows



- Matches key fields

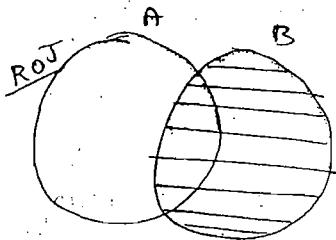
& gets the common part  
in case of Inner Join.

LoJ:



- Matches key fields

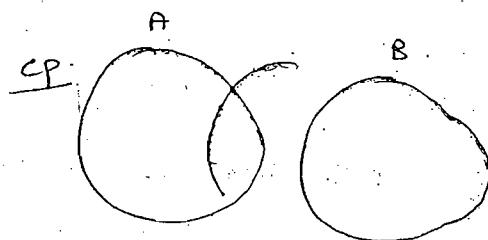
& gets common data as  
additional  
well as data in left  
table.



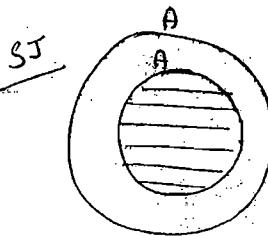
common data

+  
additional rows data

on RHS table



$10 \times 5 = 50$



Joining same table  
is called Self Join

→ we can't perform Right outer Join in  
people soft. If we still want to perform it,  
pick the table & place it on Left hand side.

## ⑥ Select with SubQuery :

The o/p of one select stmt acts as i/p for another select stmt.

## ⑦ Select with UNION :

Used to combine output of multiple select statements

## ⑧ Select with Expressions :

Used to perform mathematical operations (or) string operations.

## ⑨ Select with Aggregate Functions :

Min, max, count, sum, Avg are all aggregate functions.

## ⑩ Select with GroupBy :

Used to group a set of data & get some results.

## Ps Query Manager Accessing Methods :

### ① As-Item : (used by Developers)

Accessing in As

Open App' Designer → Go → Select Query Manager.

2-

T

[E

C

E

A

B

C

D

E

F

G

H

② n-Tier: (used by end users)

i/p for

Reporting Tools → Query

- Query manager (used to create & execute queries)
- Query viewer (used to execute queries)

2-Tier:

App' Designer → Go

↳ Query

This opens Query manager window.

**Database Tab**

↳ used to view all the list of tables from peoplesoft DB.

**Query Tab**

↳ we can see only the list of tables which are used to design select stmt.

**Fields Tab**

↳ used to view the selected list of fields from tables.

**Criteria Tab**

↳ used to create or view where conditions

### **SQL Tab**

↳ used to view sys. generated SQL statements.

### **Results Tab**

↳ used to execute & view the results of Select stmt.

Eg: PS-PO-HDR — Simple Select :

↳ Select

Create New Query

File → New

Select a record (po-HDR)

Rt. click on it.

↳ Add Record

Now we can see all the fields in Query Tab.

Select the fields

Drag & Drop (or) Double click on req. field

(or) Rt. click → select

Then **SQL Tab**

↳ consists of SQL statement with these selected fields.

**Run Query** will execute the query

To get data onto excel sheet click on icon

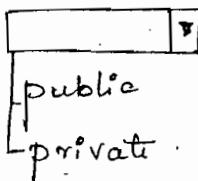
To delete a row :

Double click on that particular row

Make column as 0.

Save Query As

Owner :



Query Types :

a) Query / user query / AD HOC Query :



Normal select starts what we create which are used for reporting purpose.

b) Role Query :



This is used in workflow tool to identify the approval.

c) Process Query :



This is used for crystal reports & in vision reporting

d) Archive Query :



This is used for Archive data tool.

Archive data.

↳ Mainly used to take the backup of old data.

Query Name : BAS-Simple-21

Description :

This is a simple select

Select with Distinct :-

PO-LINE

↳ BU, PO-ID, cancel-status

File → New

Select PO-LINE

↳ Right-click

↳ Add Record

Select the fields :

↳ BU, PO-ID, cancel-status



→ This consists of duplicates.

↳ Run Query

Save Query.

↳ BAS-Distinct

Properties



Public



Distinct

Type

User Query

OK

8/12/08

USE

①

SC

C

G

CT

CL

Thi

Again Run Query

we will not get duplicate values.

Select with Order By :-

PO-HDR

↳ BU, PO-ID, PO-status

Select fields

↳ Run Query.

The o/p is ordered first in BU, But the client asks to order it by PO-ID

PO-ID field

↳ Double click (or) Rt-click.

↳ Go to **Order By** Tab.

Order by No :	1.
<input type="checkbox"/> Desc	
OK	

8/12/08

using where condition

① Select person table

where refexpr  Rexpr

↳ Select the reg. fields.

Operator

Go to **Criteria** Tab

↑  
click on  icon

↓  
This inserts empty row

Double click on Expr:1

R:

↳ select the req. field Name.

" " " operator

↳ select '='.

" " " expr

↳ either give the const. value.

Run the Query.

② we can apply where cond. by Query Tab.

Rt-click on empid

↳ select New criteria.

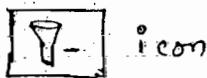


Give value for expr:1

Run the Query.

R:

To remove the criteria



icon

Select

Po.

Po.

Shift

Select

Select

Select

Select

Select

③ go To fields Tab.

↳ select the field : (empid)

↳ Rt-click

↳ New criteria



Give value for expr:1

Select with prompts:

Select person record

↳ select req. fields

Rt. click on Birthcountry

↳ New criteria

↓

Rt. click on expr

↳ prompt

This opens a window

Run-time prompt	
Field :	Birthcountry
Edit type	prompt table
prompt table :	Country-TBL

Run the Query

Select with Joins :

PO-HDR — 3 fields

K K → Parent rec  
BU, PO-ID

PO-LINE — 4 fields

K K ↑  
BU, PO-ID, PO-LINEID

Should have common key fields

↳ child rec

Select PO-HDR

↳ Select req. fields

Select PO-LINE

↳ Rt. click

↳ New Join

↑

① std Join

② Left outer Join

[OK] ↵

This opens

OK

Select the req. fields of PO-LINE Table

SQL Tab

Run the Query.

For Self Join → Select the same table @ times.

For Cartesian Join → Delete the where conditions after selecting the std. Join type.

Select with SubQuery:

PO-HDR

BUS-UNIT-TBL-FS → Select BU which are starting with 'A' & get the details of that BU from

PO-HDR

Select BU, PO-ID, PO-status From PS-PO-HDR

where BU IN (=

Select BU From PS-BUS-UNIT-TBL-FS

where BU Like 'A%'.

Select PO-HDR

↳ Select fields.

Select

To

Cond



Rt-click on BU

↳ New criteria.



Rt-click on expr

↳ Select subquery.

Select BUS-UNIT-TBL-FS

↳ Add Record.

Select BU:

↳ Rt-click

↳ New criteria

operator → LPK.

To move to Top level query.

Return to Parent



criteria → operator (IN)

To move to Sub Query

expr → subquery

→ Double click

→ Subquery always stays on right expression of where condition.

Select with unions.

To perform unions we have to follow some conditions:

a) The no. of fields selected in all the select stmts  
should be same.

b) The data type & length of the fields selected  
in all the select stmts should be same in  
the same sequence provided.

Select F1, F2, F3 From <Table1>  
[where <cond>] → char(40)

UNION

Select F1, F2, F4 From <Table2>  
[where <cond>] → char(40)  
→ <40 →  
→ >40 X

9/12/08

Eq:  
Select BU, PO-ID, PO-status from PS-PO-HDR

where BU = 'AUS01'

UNION

Select BU, PO-ID, PO-status from PS-PO-HDR

where BU = 'UK001'

To apply union

Rt click on any field → New union



→ New union icon.

To move from one select stmt to another select  
stmt -

click on 'Selection' in Query Tab:

Select Stmt

Select with expressions:

PO-LINE-DISTRIB

Selected

Time in

① To create expression

Query Tab → Expressions

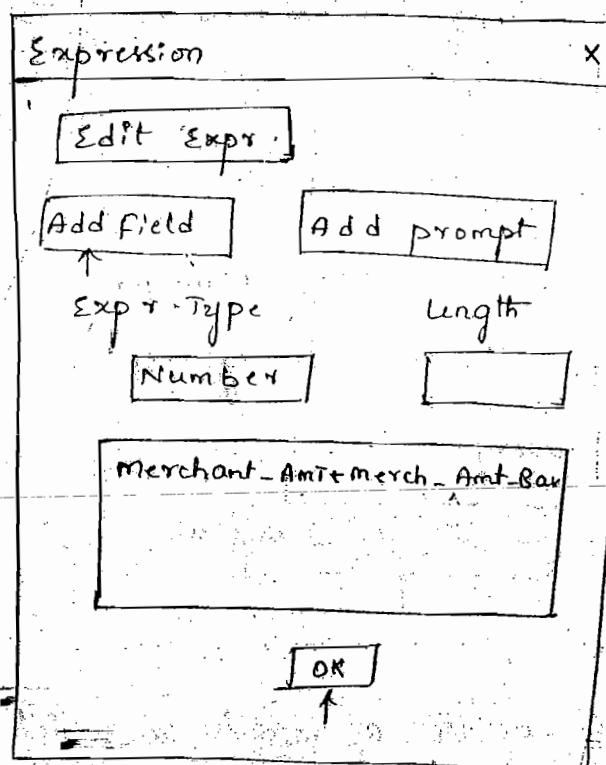
↳ Rt.click

↳ New Expression

Suppose merchant amount = Nbr, 7.2

Merchant Base Amount = Nbr, 7.2

Sum of these two = Nbr = 8.2



Go to expr → Rt.click

↳ New criteria

Operator Expr

> 1,00,000

(2) To do string operations

VENDOR\_ADDRESS

Go to expressions → Right click.

↳ New exprs:

Address1 + ',' + Address2 + ',' + state + ',' + ...

Go to fields Tab

↳ click on reg. field (addr1+addr2)  
↳ Heading  
↓  
Give Vendor address.

To change  
the column  
heading.

N-tier:

Reporting Tools → Query → Query Manager

To create New Query

click on **Create New Query Tab**.

**Records Tab**

↳ used to view list of records on which select  
stmt is being created.

**Query Tab**

→ used to view list of records from psob.

**Expressions Tab**

↳ used to create expressions

### Prompts Tab

↳ used for runtime parameters

### Fields Tab

↳ used to view list of fields which are used for select stmt

### Criteria Tab

↳ used for where cond.

### Having Tab

↳ used for Group By

### View SQL Tab

↳ used to view sys. generated SQL stmt

### Run Tab

↳ used to run & view results.

Eq: Simple Select

Records

↳ PO-HDR

### Search

Add record



It moves to Query page

↳ Select the req. fields

View SQL

↳ SQL created will be seen here.

Run the query.

Select with Distinct :-

PO-LINE

To apply Distinct

First save page query.

Go to any page, except run page

Save

Query : B25-DISTINCT-NT

Description : Testing

Folder :

Query Type : user

Owner : public

To access properties

↳ click on properties

Select Distinct

OK

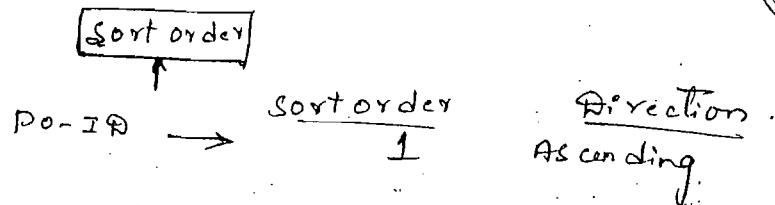
Select with Order By

PO-HDR

↳ Add record

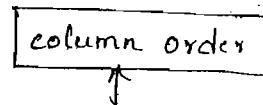
BU, PO-ID, PO-STATUS

Go to Fields page



To change column order.

Go to Fields page



Where condition :

PO-HDR → Add record

↳ BU, PO-ID, PO-STATUS

To apply where cond..

① Go to **Criteria** Tab

↳ Add criteria

Expr 1.

Business-unit

Cond. Type

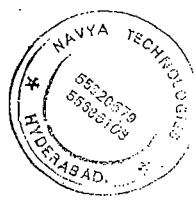
equal to

Expr 2

Avg01

② Go to **Fields** page

BU → Add criteria



Expr2 → AUS01

③ Go to Query page

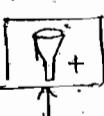
BU → 

Expr2 → AUS01

10/12/08

Select with prompt:

Go to Query page

↳ BU 

Rt. expr

① prompt

↳ New prompt

OK ↴

Select with Joins:

PO-HDR

↳ Select BU, PO-IR, PO-status.

Records Tab

↳ PO-LINE Join Record

① std Join

② Self Join

PO-HDR

Add criteria

Select fields

↳ LINE-Nbr, cancel-status.

Select with SubQuery:

BUS-UNIT-TBL-FS.

Subquery stays on RT Expr.

PO-HDR.

↳ Select BU, PO-ID, PO-STATUS.

Fields Tab

↳ Add criteria

RT Expr. → @ Subquery.

operator - in list.

Define Sub Query

Select BDS-UNIT-TBL-FS.

↳ Select BU field.

Add criteria.

Operator - Inlist.

RT Expr. - A%.

To Navigate for parent query or Subquery

Subquery/Union Navigation

Select with Union :

1<sup>st</sup> PO-HAR.

↳ BU, PO-ID, po-status

Criteria BU = A0501.

To apply union click on New union.

2<sup>nd</sup> PO-HAR.

↳ BU, PO-ID, po-status

Criteria BU = V5001.

To move from one select stat. to another select stat.

click on Subquery/Union Navigation.

Select with Expressions :

PO-LINE-DISTRIB.

① ↳ BU, PO-ID, Mex-Amt, Mex-Amt-Bare

To create Expressions

Go to [Expr] page

Add Expr

Expr Type : INBY

length : 8

Decimals : 2

Expr. Text

Mex-Amt + Mex-Amt-Bare

Add Fields

use as Field



Go to Expr → Add criteria



operator : >

Rt-expr : 1,00,000.

(a) string operation.

VENDOR\_ADDRESS

↳ Vendor-id, Name

Go to Expr page.

Expr-Type : character

length : 14 ♀.

Expr.Text :

Address + ',' + Address + ',' + City + ',' +  
State + ',' + Country

Add Fields



use as Field



Run the Query.

To change the heading

Edit (click Edit on req. field)



Heading : Address

## 2-Tier :

Eq:

Effective Date : (Vendor-addr)

How to create effective dated Query ?

① Effective Date  $\leq$  @currentDate

Get rows which are get effected as of today date.

② eff.Date  $\geq$  @currentDate

↳ get current as well as future rows

③ eff.Date < @currentDate

↳ get history rows only

④ eff.Date > @currentDate

↳ get future rows only

⑤ First effective date

↳ gets most history rows

⑥ Last eff. date

↳ max. future rows

⑦ NO eff. Date option

## N-tier :

Vendor-addr

Change the value as per requirement in  
Criteria Tab.

Eq: B25 - Query-Tbl

↳ Emplid, Name, comprate  
↓  
key.

Build the Table.

Insert some data.

Go To Query mgs.

↳ B25 - Query-Tbl

\* To access the tables which are designed in App' designer using Query manager . we have to follow some steps

- ① Create a New Query Tree & add the record (or) add the record to an existing tree.
- ② Add the Query Tree to permission list (if not added)

Already existing Tree - NO Need to add to PL

- ③ Run Query Access List cache process

Create a New Query Tree :-

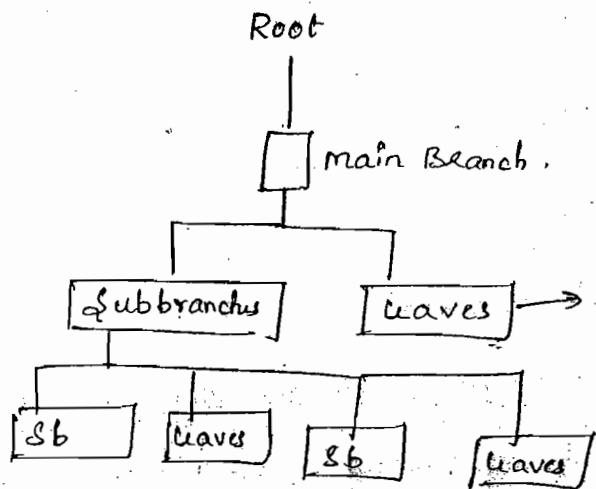
People Tools → Security → Query Security

↓  
Query Access Manager

Main Branches - Access groups  
leaves - Records

Aff

Ad.



are ending pieces -

Ac

Q

To Re

P

Set

Q

Now

Adding

PT

Create a New Tree

✓ Tree Name : B25 - Query-tree

str. ID : Access-group

✓ Descr : B25 Test Tree

OK

16/12/08  
Add main Branch

Access Group : Accounting

Save

Accounting - Accounting

To add sub branches click on insert child rec.

Select B25 - Dept-Tbl

Add

B25 - Depen-Tbl

After adding all records click on Save ↗

Adding to PL: PT → Security → Permissions & Roles

Add. To AFAS1000

go to query page

↓  
Permission lists

Access group permissions ↗

Add QueryTree we created to it. [OK] ↳

[Save] ↳ the page

To Run the Query Access List cache:

PT → Security → Query security

↓

Query Access List cache

Select ① Enable Access List cache.

[Run] ↳

See in the process Monitor

[Refresh] ↳

Now see the Query in Query Manager.

Adding Records To Existing Tree:

PT → Security → QueryTree Security

↳ Query Access manager

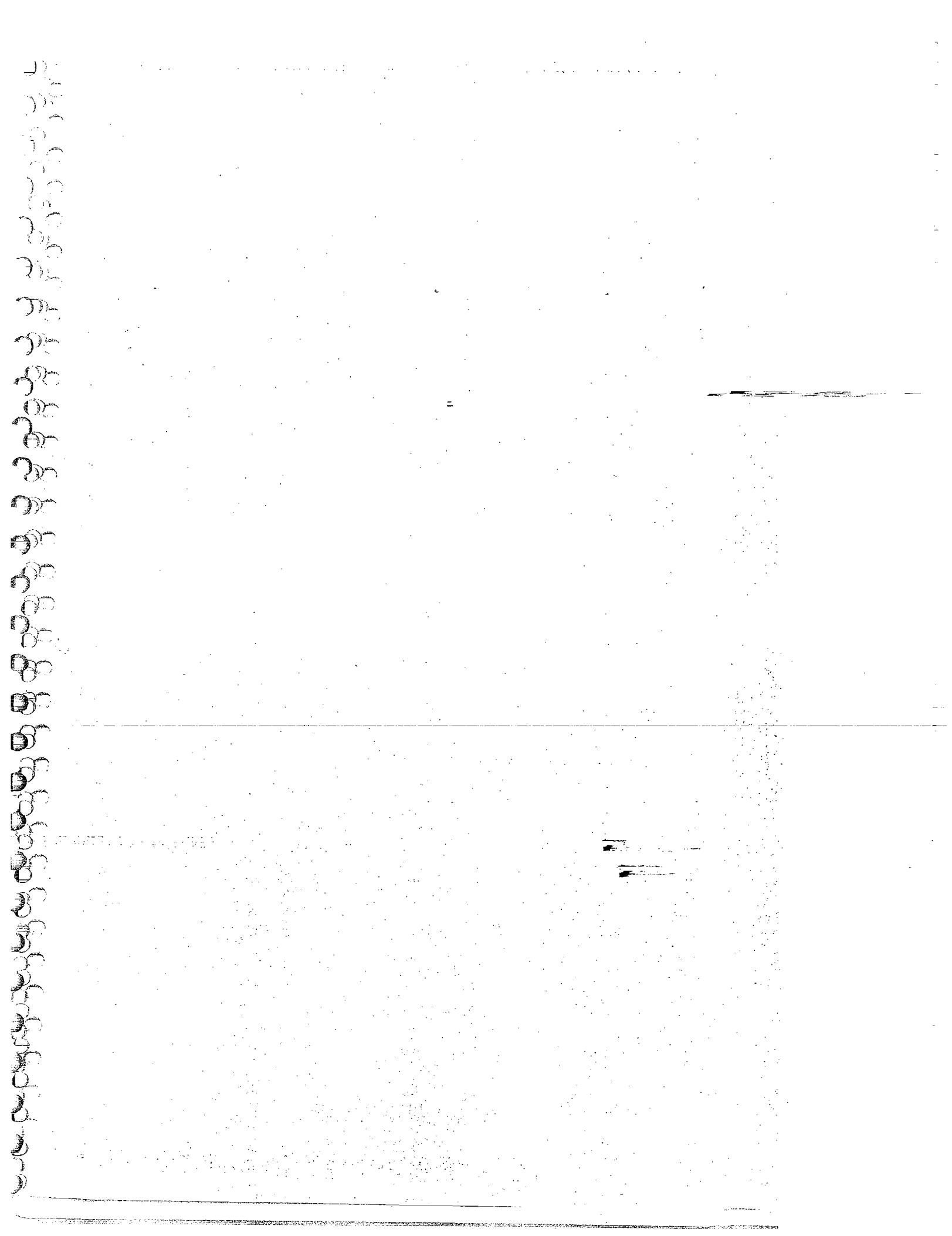
## Basic Search

Search By:

Tree Name:  ↪ B25\_Query\_Tree ↴

Takes to the already existing tree page

→ Select the insert child record icon & add  
the reg. table to it.



1000 900 800 700 600 500 400 300 200 100

13/12/08

## People Code



- programming language in ps.
- Buffer in ps is component Buffer.

### Advantages:

- Event driven programming lang.
  - Depending upon the action that have occurred the code gets executed.
- used for validations.
- used to enter data programmatically.
- used to implement Business Logic.
- used to enable Navigations
- used to update DB & Buffer
- used to reuse code
- used in integration tools.
- User friendly.
  - ↳ This will have a editor which helps to send the syntax errors while writing code.

### Available Event Types:

- The location where we write people code is Event types.

Record — 15 locations

Page — 01 "

Component — 02 "

John C. Dill

## Menu

— 01 locations

Comp. interface

— 01 "

App' Engine prgm

— 01 "

Message

— 04 "

## Record Events:

### A) Field level

### B) Search level

- 1) Field default
- 2) Field Edit
- 3) Field change
- 4) Field Formula
- 5) Pre popup

- 1) Search Init

- 2) Search Save

### C) Row level

### D) Save level

- 1) Row Init

- 1) Save Edit

- 2) Row Select

- 2) Save prechange

- 3) Row Insert

- 3) WORKFLOW

- 4) Row Delete

- 4) Save postchange

## Component Events:

\*Events specific to comp are preBuild & postBuild.

### A) Component Level

### B) Comp Record field Level

- 1) pre Build
- 2) post Build
- 3) Save pre change
- 4) work flow
- 5) Save post change

- 1) Field Default

- 2) Field Edit

- 3) Field change

- 4) pre popup

### c) comp. Record level

- 1) Search Init
- 2) Row Delete
- 3) Search Save
- 4) Save Edit
- 5) Row Init
- 6) Save prechange
- 4) Row Select
- 7) Save postchange
- 5) Row Insert

### Page Events :

- 1) Activate

### Menu Events

- 1) Item Selected

### App' engine Events

- 1) on Execute

### CI Events

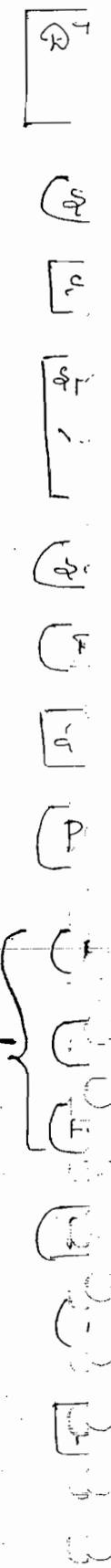
- 1) Methods

### Message Events

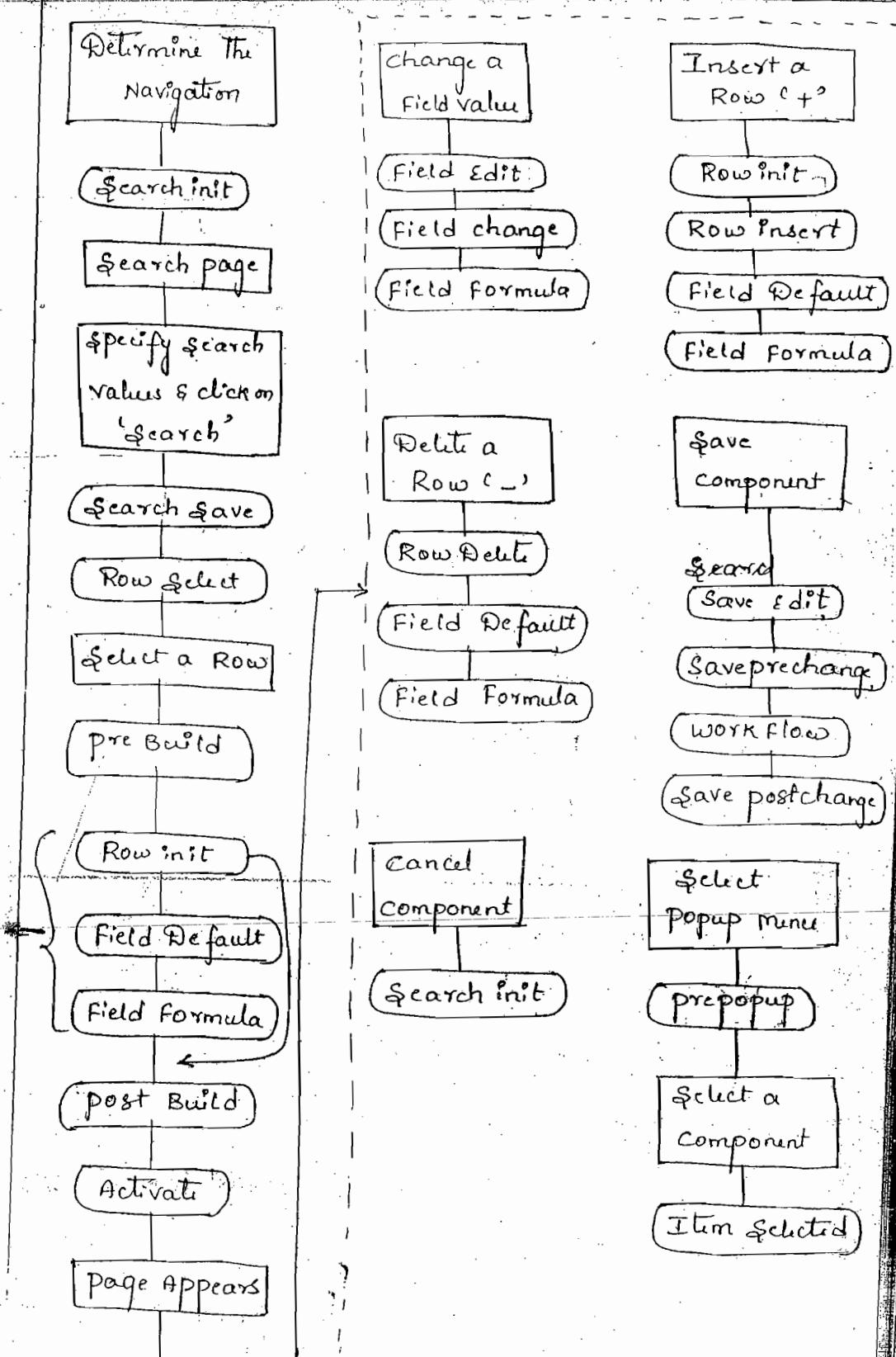
- 1) On Request
- 3) On Route Receive
- 2) Subscription
- 4) On Route Send

### (I|R) Event Execution Sequence:

\* comp.  
Build  
Events



Comp.  
Build  
Events



## EVENTS :

### Search Init :

- performs before the search record dialog box appears.  
(Search page)
- provides a way to control processing before an operator enters values in the search keys & alt. search keys.
- people code is attached to the search key or alt. search key.
- used to enforce Row level security by restricting the user from changing the search values.
- important func's used
  - a) `SetSearchDialogBehaviour`
  - b) `SetSearchEdit`
  - c) `SetSearchDefault`
  - d) `ClearSearchDefault`

### Search Save :

- performs when the operator clicks 'OK' in the Search Dialog box.
- provides a way to edit the information entered in the search record dialog box.
- used to force the user to insert atleast one → All

Row

→

→

→

→

→

Pre

→

→

→

→

Field

→

field into the dialog box even if it is a partial value.

### Row Select:

- performs as the app processor reads data into the component.
- used to filter out rows of data as they are being read into the component buffer.
- Rarely used bcoz it is inefficient to filter out rows of data, after they have already been selected.
- Instead of this we will be using views for Row level security
- Important func's are
  - a) Discard Row
  - b) & Top Fetching.



### Pre Build:

- performs before all the comp. Build events
- Use:
  - a) Hide & unhide pages
  - b) Set global & component variables
  - c) Restrict some users to access the component

### Field Default:

- Attempt to set defaults of fields without a value.

we can perform this using Record fld properties

↓  
Constant [ ]

But only one value we can populate as default.

If we want to set state as city as 'Hyd' for dept 10 & city as 'Bang' for dept 20. Then we use this Field Default.

→ Record Field properties overrides this code.

→ This performs only when the field to which it is attached has no values.

(Record field prop. has higher priority)

#### Field Formula :

→ This executes after Field Default completes successfully.

→ Every user action executes Field Formula people code (Except Save action)

→ This is used in Derived/Work Record to define func. libraries.

#### Row init :

→ Executes when the processor encounters a Row of data for the first time.

→ Executes for every new row of data into buffer

Properties

- Executes when the user inserts a new row.
- Used to assign calculated values to the fields.
- Used to change the properties of other fields based on one (or) more field values.
- 90% of field change code can be seen in this event.

### Post Build:

- performs after the Comp. Build events have performed.
- used to hide & unhide pages as well as to set page variables
- This is associated with comp.
- used to cal. values & set display characteristics of an object
- Code placed in RowInit is often placed here.

### Activate:

- Executed whenever a page is activated (or) appeared
- Every page has its own Activate Event
- Effectively eliminates the need to write page specific Row init code

Eg: If page 1, hide city field, page 2, hide country field etc.

For such type of cases use this event

- used for security validations like page hiding.
- Associated with pages.

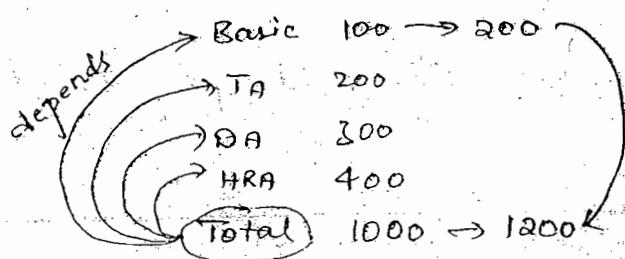
### Field Edit:

- Performs after the Field value has been changed and the new value of the field satisfies the std. system edits. (Eg: Req. Resonabldat, prompt table edit)
- used to validate the content of the field.
- Only fires on a specific field (or) row that just changed.
- Associated with Record fields & comp. record fields.

### Field Change:

- performs once the field has changed, the std. sys. edits & field edit prologue has completed successfully.
- used to perform additional processing that is based on the new value of the changed field

#### EID



- used to assign cal. value to the other fields based on New value.
- Not used for validations
- Rowinit & Field change are complementary events

#### Row Insert :

- Performs when a new row of data is added (F7) within a scroll.
- Rowinit peoplecode is also executed when a new row is inserted.
- used to restrict some users to insert rows into scrolls.
- Associated with Record Fields & comp.Record peoplecode.

#### Row Delete :

- Performs when the row is deleted (F8).
- Used to cal. Running Totals & to prevent a row from being deleted.

Bill No : ~~Bill Date~~

1. Rice 5kg 150
  2. water 10lt 200 → If deleted.
  3. sugar 5kgs 500
- Newly added

Total : 350.

↳ 850 ↳ 650

→ After row delete finishes, the app' processor  
proceeds through Field Default & Field Formula  
peoplecode

→ Associated with record fields & comp. records

### Save Edit :

→ Performs once the operator tries to save the  
parent group (or) comp.

→ Used to validate data before the data is  
updated on the db.

→ The comp. processor applies SaveEdit peoplecode  
to all non-deleted rows of data in the  
buffers & all pages in a comp.

→ SaveEdit is better suited than Field Edit  
if the validation references more than one field  
or row of data.

→ Associated with record fields & components.

→ Important func:

a) setCursorPosition.

### Save Prechange :

→ Executes after SaveEdit performs successfully.

→ This provides one last chance to manipulate  
data before the db is updated.

→ used to change display characteristics & validations.

→ Associated with Record Field, comp & comp record

### Work Flow:

→ used to segregate people code related to workflow from the rest of ur app.

→ Associated with Record Fields & components

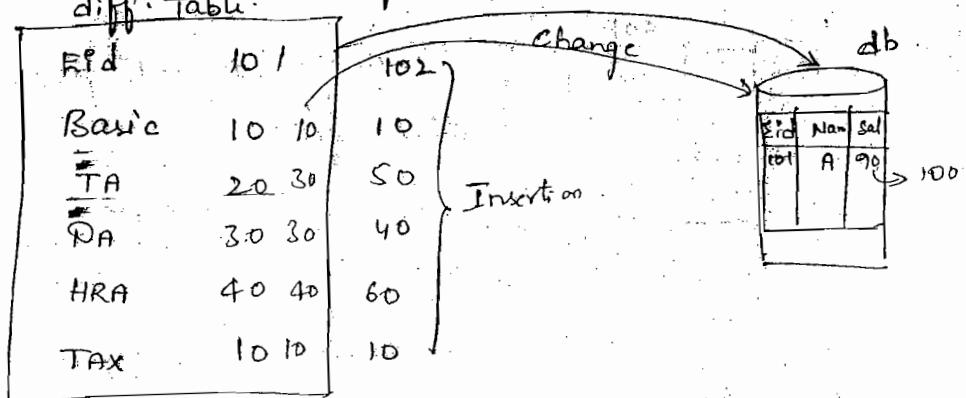
### Imp. function:

a) Trigger Business Event

### Save Postchange:

→ performs after save prechange & work flow completes successfully.

→ used to insert (or) update values of the record fields in the db that are not available on comp. buffer diff. Table.



→ Associated with record fields, comp & Comp. record.

### Imp. function : SALEXEC

### On Execute:

- Used to write people code programming in AF prgm.
- Associated with AF prgm.

### Method:

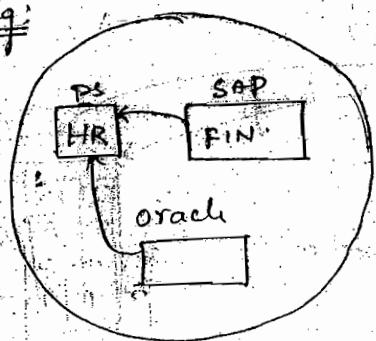
- Used to write people code programming on CI.
- Associated with Component Interface Integration tool.

### Subscription:

Getting data into ps → subscribe }

Sending data out of ps → publish }

Eg:



Receiving data into ps sys. from other systems.

- Used to validate the message to subscribe which is sent by (or) published by other sys.
- Used in messaging & Integration Broker.

### On

- U.
- by
- Us.
- Pr.

### On Rot

- 
- 8
- U.

### On Rec

- U.
- Th.
- a)

Or

me

b)

PS

IS

re

ce

de

re

re

Route  
On Request Send :

- Used to specify programmatically to which sys. we have to publish the message.
- Used in Messaging (or) Integration Broker integration tool.

OnRoute Receive :

- Used to programmatically decide from which sys. we want to subscribe.
- Used in Messaging (or) IB. Integration tool.

OnRequest :

- Used to request for a message programmatically
  - This works in 2 ways
    - a) Synchronised : Online, that is if the other sys. is down it shows error msg. means the messaging fails. Eg: chatting
    - b) Asynchronous : offline (Non-Real time), that is even though the sys. is down the message is sent.
- Eg: Messaging

## Navigations for People code :

### Three Navigations :

#### 1) Designer window

For comp. we cannot get this.

Open Record → Rt. click on field



View people code .

#### 2) Menu Bar

Open reg. comp → Go to view



View people code .

#### 3) Project workspace

Insert reg object into project



Rt. click on it



View people code .

## People code language :

### Imp. Elements

#### 1) Data Types

#### 5) Operators & separators

#### 2) comments

#### 6) programming constructs

#### 3) statements

##### i) Assignment

##### 7) System variables

##### ii) Defining

##### 8) Object oriented classes

##### iii) Declaration

#### 4) Built in Functions

Cont

→

→

→

→

→

→

→ T.

→ Each & Every people code stmt ends with ;?

### Data Types:

#### A) Conventional Data Type

String

Number

Date

Time

Date Time

Boolean

Object

Any → Default

Float

Integer

#### B) Object Oriented Data Type

Field

Record

Row, Rowset

Grid

Grid column

Page

File

Array

Message

SQL

→ conventional data types used in procedural lang.

→ object oriented " " " " object oriented lang.

### Commenting:

→ For Single Line

Rem/Remark .....

→ For Multiple Line

/\* ..... \*/;

→ Text that will not be executed even when we write in people code editor.

## Scope of Variable :

Variable — Temporary memory location that holds  
the value

→ lifetime of variable.

1) Global : valid for the entire session

Signin to signout is called session.

2) Component : valid while any page in the comp. in  
which it's defined stays active.

3) Local : valid for the life of the people code  
program or func. in which it's defined.

4) Page : valid for the page.

## Declaration :

Specifying the scope & datatype of variable

i.e used to specify state of variable.

### Syntax:

<scope> <datatype> & <variable Name>

System variables : The values of

variables which will be maintained by the

System.

Starts with %.

Declaration not req.

eg: %userid, %component, %page etc.

### User Defined Variables:

Variables which are defined by developers  
starts with '&'

- i) Local: Not required for local variables  
(Default scope)

Eg: Local Number & Age;

- ii) Global: Declaration req.

Eg: Global String & Oper-Nickname;

- iii) Component: Declaration req.

Eg: Component Rowset & my-Rowset

### Assignment / Defining statement:

#### Syntax:

Expression = Record field / Variable / constant /

function / combination of above values with  
mathematical operators ;

= Record field / Variable = Expression ;

Eg: &temp = 10 ;

&temp = 'Ravi,s' ;

&TotalSal = &Basic + &TA + &DA + &HRA ;

## System variables:

1) % AsOfDate

↳ stores the current date as per the application

2) % clientDate

↳ To get data as

3) % component

holds the particular comp. Name

4) % compIntfcName

stores the current CI name

5) % DateTime

stores date & Time value

6) % dBName

returns db Name

7) % employeeId

Returns the empid of person who is currently logged in to sys.

8) % menu

Returns the current menu Name of the current page we are seeing.

9) % mode I R

% mode returns a string value consisting of

an uppercase character specifying the action a user selected when starting the current comp.

The foll. values can be returned.

A - Add

V - update / Display

L - update / Display All

C - correction

Eg:

Name	<input type="text"/>
Eid	<input type="text"/>
Sal	<input type="text"/>

(If we logged in to system with Add mode).

If we want to add a new value then the

Sal field must be not Display only, while

Seeing existing data the Sal field must be

Display only. In such cases we write code in  
'PreBuild'.

Eg: If mode <= 'A' Then

    make salary as Display only

End-If;

10) %operatorid (or) %Userid  
↳ prev. version  
(returns the current user logged)

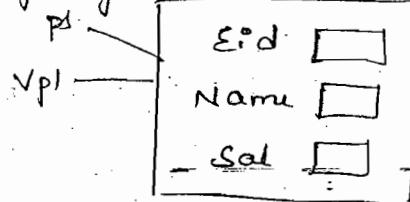
Eg:

ps - seeing existing data

Sal - Display only.

Vpl - Editable.

write code in pre-build



→ we can write user specific coding

Eg: If %mode < 'A' And %operatorid = 'ps' Then

    make salary as display only

End-if;

11) %Page

Returns Name of page in uppercase

12) %PermissionList

returns all pl's that a current user can have

13) %Role

returns all the roles the current user has

Functions: (1)

There are 4 types of functions:

1) Built-in Functions

2) Internal peoplecode Functions

3) External peoplecode Functions

#### 4) Ext. Nonpeoplecode functions.

→ piece of code that is reused is nothing but a function.

##### Advantage of Functions:

→ Reusability.

→ Ease of maintenance.

→ Every function will have 3 parts:

i) Definition - place where we assign name to a set of lines of code.

ii) Declaration

iii) calling.

Definition: place where source code (reusable code) is assigned to a function.

Declaration: used to specify the location of definition.

Calling: Execution of Function.

#### Built In functions:

→ These are peoplecode delivered functions.

→ This does not require declaration, we can directly use the functions.

Eg: Hide (<fieldName>);

SALLEXEC (<sqlString>);

## 2) Internal peoplecode Functions:

→ we call the func. at the same place (or) prgm,  
where we define.

→ Declaration is not required.

Fun

## 3) External peoplecode Functions:

→ These functions are defined as Functions

libraries. (Defined (or) written in the Derived/  
work Record)

→ Declaration is required.

→

## 4) External Non-peoplecode Functions:

→ These functions are defined under the external/  
Third party systems libraries

→ Declaration is required.

### Function Declaration : (2)

→ External peoplecode function

Syntax:

Declare function <fun.name>

    Peoplecode <rec.name> <fieldName.eventType>

→ External Non-peoplecode Function

Syntax:

Declare function <fun.name> Library

    <library.name> <parameters>

→ Func. Declaration is not req. for build in  
prgm. func's & internal people code functions.  
so Syntax is not available.

### Function Definition: ①

→ Function def. is of three types

i) Function Not Returns:

Syntax:

Function fun.name (arguments)

statements;

End-function;

ii) Function Returns:

Syntax:

Function fun.name (arguments) Return

<datatype>

statements;

End-function;

iii) Function Optionally Returns:

Syntax:

Function fun.name (arguments) Return Boolean

Statements;

End-function;

P.

i

ii

iii

iv

v

vi

vii

viii

vix

vixi

vixii

vixiii

vixiv

vixv

## Function call:

→ Function calling is of 3 types.

i) Function Not Returns :

Syntax:

function name ( $\downarrow$ ) ; parameters

ii) Function Returns :

Syntax:

<Variable> = function name ( $\downarrow$ ) ; parameters

iii) Function optionally Returns :

Syntax:

<Variable> = function name ( $\downarrow$ ) ; parameters

Always the datatype of this variable is 'Boolean'.

## Operators:

→ Operators are of 3 types

i) Mathematical operators

Eg: +, -, \*, /, % etc.

ii) Relational operators

Eg: =,  $\neq$ ,  $\geq$ ,  $\leq$ ,  $>$ ,  $<$ ,  $\geq$ ,  $\leq$ ,  $\neq$  etc.

↓  
Not equal to

iii) Boolean operators

Eg: AND, OR, NOT etc

## Programming constructs:

### i) IF :

→ This is used to check the conditions

Syntax:

```
If condition Then  
    statements;  
Else  
    statements;  
End-if;
```

### ii) Evaluate :

→ This is same as the switch statement in C.

→ Used to check the multiple conditions.

Syntax:

```
Evaluate < field (or) variable >  
when = < value >  
    statements;  
Break;  
when other  
    statements;  
End-Evaluate;
```

Eg: Evaluate & Peptid

→ Break;

when = 10

=

Break;

when = 20

=

when = 30

=

Break;

when - other

=

End-Evaluate

### iii) FOR:

- This is a looping construct.
- Used when the ending cond. is known.

Syntax:

For <variable> = <start value> To <end value>  
[<step value>]

statements ;

End - For;

- If we don't specify step value, by default var. value increments by 1.

### iv) WHILE:

- This is a looping construct.

- Used when the ending condition is not known
- The check for the cond. is performed before entering the loop.

Syntax:

While <condition>

Statements ;

End - while;

### v) Repeat Until:

- This is a looping construct.

- Enters the loop and check for the cond. after that.

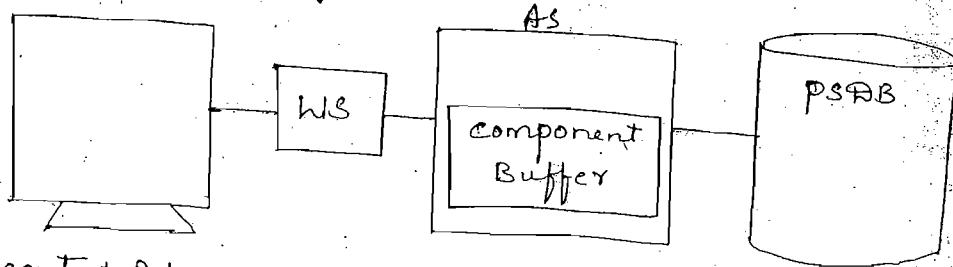
Syntax: Repeat  
statements ;  
Until <cond>;

→ If cond. is satisfied there is no diff b/w while & Repeat until.

→ If cond. is not satisfied first time itself the code will not get executed in while but in Repeat until the code gets executed atleast once.

### Component Buffer:

↳ Temp. memory loc. where data is stored.



content Ref.

emplid 10

Name

103 e

flags

ADD Delta change

Search

- 101 A
- 102 B
- 103 C
- 104 D
- 105 E

103	e	10	50	(20)	30	HD
1001	x	21				
1002	y	(22)	32			
1003	z	45				
1004	g	54	7			

## Component Buffer Functions :

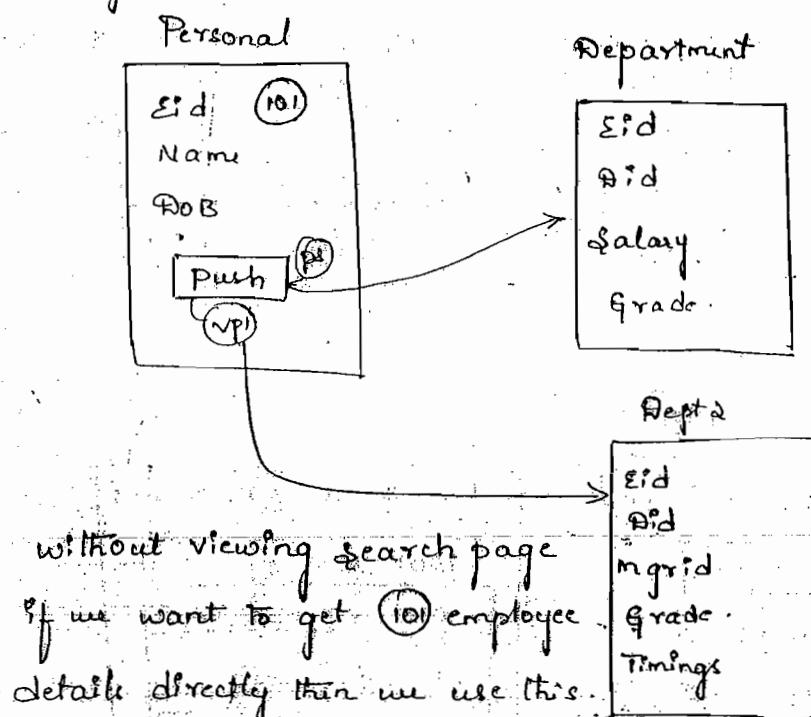
### 1) Active Row count :

Syntax :

Active Row count (\$ scroll path)

### 2) Add Key List Item :

Eg:



without viewing search page

If we want to get (101) employee details directly then we use this.

Syntax :

Add Key List Item (field, value)

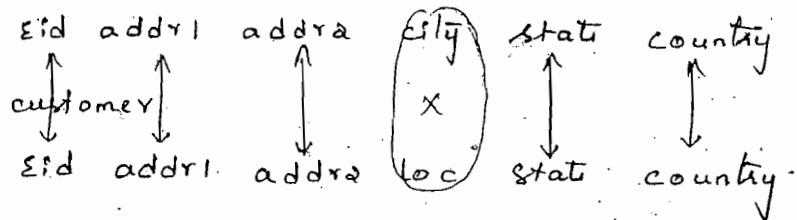
→ Used to add a field to a keylist & then transfer to a page which uses that field as a key.

→ Returns a Boolean Value indicating whether it completed successfully.

### 3) CompareLikeFields :

Eg:

personal



→ This compares only like name fields.

→ Returns True or False.

If value matches      If values does not match

Syntax :

CompareLikeFields (from, to)

From/To : level, & scrollpath, target-row

### 4) ComponentChanged :

→ Determines whether a comp. has changed since the last save, whether by the user or by people code.

Eg: ps - go in, save data, can change data

vpl - "", "", "", cannot = " of any field

If he want to change & save data then we have to thro' error error.

The code is need to be write in 'SaveEdit'.

→ Returns True / False

Code in SaveEdit is

If %operatorid = 'VPI' AND  
Componentchanged() Then  
    Error message;     ↳ if we not specify anything  
                                means = True.  
End-if;

Syntax:

Componentchanged()

Eg:

5) CopyFields:

Copies like-named fields data.

Syntax:

CopyFields (from, to)

From / To = Level, scrollpath, target-row

6) CurrentLevelNumber:

Returns level of particular record on that particular page.

Syntax:

CurrentLevelNumber()

Eg: &level = CurrentLevelNumber()

7) CurrentRowNumber:

Returns the no. of particular row on that particular scroll.

Syntax:

currentRowNumber()

8) DeleteRow:

Delete row programmatically.

Syntax:

DeleteRow(scrollpath, target-row)

→ we must write the code in parent level, but  
we should not write in same level or child level.

9) DiscardRow: (I)

This prevents a row from being added to a  
page scroll during RowSelect processing.

Valid only in RowSelect people code.

Syntax:

DiscardRow()

10) Fetchvalue:

Returns the value of a buffer in a specific  
row of a scroll level

Syntax:

Fetchvalue(scrollpath, target-row, [recordname],  
fieldName)

11) Recordchanged:

Pragmatically verify whether the data in record  
has changed (or) not.

True — any field value in par. record changed.

False — No change occurs.

Syntax:

Recordchanged (&scrollpath, target-row)

12) ScrollFlush:

To Remove all rows of data for that particular scroll from the comp. buffer.

13) ScrollSelect:

To get data from db back to comp. buffer.

Syntax:

ScrollFlush(&scrollpath)

ScrollSelect(&scrollpath)

Refresh = ScrollFlush + ScrollSelect

→ Rows that are flushed are not deleted from db.

This (SF) func. is often used to clear a work scroll.

14) SetCursorPosition: before a call to ScrollSelect.

→ Used to place the focus in a specified field anywhere in the current comp.

Syntax:

SetCursorPosition(Page, pagename, &scrollpath,  
target-row, [recordname.]fieldname)

Ex:

I  
D

F1.

Jf

col

exc

IT

cur

we

15) E

g

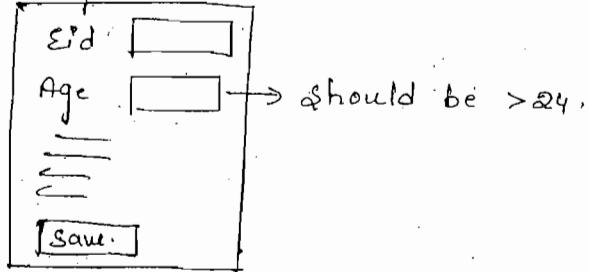
C

→ AF

thu

C

Ex: Empl-Tbl.



→ should be > 24.

If we want to validate 'Age' field write code in Field Edit:

= Local Number & Age;

& Age = Fetchvalue (Empl-Tbl. Age);

If & Age < 24 Then

Error message;

End-if.

If want to validate in no. of fields write code in 'Save Edit', that code gets executed when we click on **Save** Button.

It simply displays error, it will not move cursor to that particular pos. for such cases we use this Setcursorpos.

15) SetDefault:

Syntax:

SetDefault (recordname. ] fieldname)

→ After the Field Default is executed, it simply removes the value from the par. field.

## 16) Total Row Count : (I)

Returns total no: of rows in scroll area.

103

(including rows marked as deleted)

1001
1002
1003
1004

ARC TRC

4 4

3 4

→ If marked  
deleted.

A  
→

9

B  
D

D C

Syntax:

Total Rowcount (scrollpath)

→ TotalRowcount = Active Row count + Rows marked as deleted.

2) C

## 17) Transfer : (I)

Used to move from one page to other page

Syntax:

Transfer(new-instance, [True - displays dest. page in separate page], [Same source page])

MenuName.menuName,

BarName.barname;

= ItemName.menu-item-name,

Page-component-item-name,

action [, keylist] [, AutoSearch]);

3) Ge

Si

C

4) G

SL

G

G

## 18) % TruncateTable :

\* SQL → Truncate - Stmt cannot be rollback.

→ Delete - Data can be rollback.

~~All area.~~  
~~as deleted~~

# Truncate = Delete + commit.

→ Deletes all the rows in a table.

Syntax:

? TruncateTable (table-name)

### DataBuffer Functions:

#### 1) CreateRecord:

Syntax:

CreateRecord (Record . recname)

#### 2) CreateRowset:

Syntax:

CreateRowset ({ Record . recname ; & Rowset }

[, { Field .fieldname , Record . recname ;  
& Rowset } ] ... )

→ The CreateRowset func. creates an unpopulated  
standalone rowset.

#### 3) GetField:

Syntax:

GetField ([recname : fieldname])

#### 4) GetLevel0:

Syntax:

GetLevel0()

### 5) GetRecord :

Syntax:

GetRecord([Record, recname])

Eg: &Rec = GetRecord();

### Date/Time Functions :

#### 1) %. DateAdd :

Returns a date by adding add-days to date-from. add-days can be negative.

Syntax:

%. DateAdd (date-from, add-days)

#### 2) %. DateDiff :

Returns an integer representing the diff. b/w 2 dates in no. of days.

Syntax:

%. DateDiff (date-from, date-to)

#### 3) Date :

Used to convert no. in the form YYYYMMDD and returns a date value. If the date is invalid, Date displays an error msg.

Syntax:

Date (date-num) Format: YYYYMMDD

#### 4) DatePart:

Pickup a date value from a Datetime field.

Syntax:

DatePart ( datetime - value )

#### 5) Datavalue:

Converts a date string & returns the result as a Date type. date-str must be a string in the active date format.

Syntax:

Datavalue ( date-str )

Eg: &DTM = Datavalue ("10/09/97")

#### 6) Day:

Identifies the date field. Supports both Date &

Datetime datatypes.

Syntax:

Day ( dt-val )

#### 7) IsDate:

To determine if value is in appropriate date format or not.

Syntax:

IsDate ( value );

#### 8) Month:

Returns the month of the year.

Returns 0 - 12

Syntax:

Month (datavalue)

Eg:

No person should hire in December

C  
=

) C

Field Edit of DOJ Field.

Local Date &Dt;

&Dt = Fetchvalue(Empl-Tbl. DOJ);

a)

If month(&Dt) = 12 then

Error msg;

End-if;

9) Weekday:

Syntax:

Weekday(dt)

Eg:

No person should Join Sat & Sun

Th  
→

Fieldedit of DOJ Field.

Local Date &Dt;

&Dt = Fetchvalue (Empl-Tbl. DOJ);

If weekday(&Dt) IN (1,7) Then

Error msg;

End-if;

→ calculate the day of the week based on  
a date value.

→

→

→

→

→

→

→

→

→

→

→

## Conversion Functions :

### 1) String:

converts any Non-string datatype to string.

Syntax:

String(value);

### 2) Value:

→ converts a string representing a number to the number.

Syntax:

Value(str)

## Think-Time Functions : (I) \*

→ This functions suspend processing either until the user has taken some action (or) until an external process has run to completion.

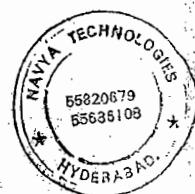
→ Think-time functions should be avoided in any of the foll. people code events:

i) Save prechange

ii) Work flow

iii) Row select

iv) Save postchange



on

→ The below are some think-time func's.

① WinMessage ② Message Box

### Syntax :

i) MessageBox (style, title, message-set, message-num,  
default-txt [, paramlist])

ii) Winmessage(message [, style] [, title]);

iii) Warning "str";

iv) Error "str";

→ Depending upon style provided, msgbox works.

Style 0 — winmessage

Style 1 — warning

Related to

Style 2 — Error

message catalog

→ If message-set & message-num is not correct  
then default txt will appear.

Eg: (B25-pcode-project)

Record, (B25-pcode-Tbl)

↳ Emplid, Name, Salary

↓

↓

K,SK ASK

Create a page (B25-pcode-Pg)

" " comp (B25-pcode-comp)

Register the comp.

Go to Emplid.

↳ View people code

(searchinst)

20/12/08  
SCE  
→  
Pc

winmessage("SearchInit - Record level");

SearchSave

winmessage("SearchSave - Record level");

Go to comp:

↳ view people code.

B25-pcode - Tbl

SearchInit

winmessage("SearchInit - CompLevel");

\* If we write code both in record level & comp level, the record level code gets executed first.

Go to Record:

↳ emplid. (SearchInit)

winmessage("%operatorid");  
" (%component");  
" (%page");  
" (%permissionlist");  
" (%mode");  
" (%asofdate");  
" (%dbname");

People Books → people code hang. ref guide.

SQLEXEC: \* (I) (R)

→ The SQLEXEC func. executes a SQL command from within a people code program by passing a SQL command string. The SQL

Command bypasses the component processor & interacts with the db server directly. If we want to delete, insert (or) update a single record (row), use the corresponding people code Record object method.

→ To delete, Insert (or) update a series of records all of the same type, we need to use SQL class.

→ SQLExec can only select a single row of data

→ If more than one row of data is retrieved by the SQLstmt only first row will be processed.

→ Syntax:

```
SQLExec({$sqlcmd},SQL.sqlname,$bindexprs,  
outputvars)
```

Eg:

```
SQLExec ("Select sum(posted-total-amt) from  
ps-ledger where deptid between :1 and :2",  
@DeptId -from, Dept-To, &sum);
```

→ can be used in

Save prechange

Workflow

Save post change

Field change.

} DML &  
Select

Other events

↳ only select

for &  
if we want  
record (row)  
3rd object

of records  
in SQL

2nd of data  
retrieved  
will be

exprs,

from

: 2

nts  
& select

→ we can write SQL exec in any event but the diff is in the mentioned 4 events we can write any DML & select stmt but in other events we can write only select stmt.

scroll path : (L) R

① Referring a Record:

• level 1

Syntax: Function(SCROLL/RECORD. <LVL1 recname>)

Eg: HideScroll (SCROLL. Empl-Tbl)

Dept-Tbl, level 0 → 10 ELcc HYd

Empl-Tbl {  
Level 1 }  
101 A 1000  
102 B 2000  
103 C 3000

Depen-Tbl {  
Level 2 }  
1001 X 21Y  
1002 Y 22Y  
2001 M 42Y  
2002 N 42Y

Edut-Tbl {  
Level 3 }  
SSC 800 B  
Degree 750 C  
B.Tech 850 A  
M.Tech 800 B

level 2

Syntax: Function (SCROLL. <LVL1 recname>,  
<LVL1 rowno>, SCROLL. <LVL2 recname>);

Eg: 1. HideScroll(scroll.empl-Tbl, 0, scroll.Depen-Tbl);

↓

Hides the dependents of 101. 103.

(2) 1

h

g

En

2. For &i=1 To ActiveRowCount(Record.empl-Tbl)

{ HideScroll(scroll.empl-Tbl, &i, scroll.Depen-Tbl); }

} END-FOR .

→ Hides the dependents of all employees.

Level 3:

Syntax: Function(scroll.<lv1 recname>, <lv1 rowno>)

scroll.<lv2 recname>, <lv2 rowno>,

scroll.<lv3 recname>);

hev

sy

Ec

Eg: 1. HideScroll(scroll.empl-Tbl, 3, scroll.Depen-Tbl,

scroll.edu-Tbl);

→ Hides the Edu of 200 a dependent.

Needs to identify the emp. id also.

he

sy

2. For &i=1 To ActiveRowCount(Record.empl-Tbl)

{ For &j=1 To ActiveRowCount(Record.Depen-Tbl,

&i, Record.Depen-Tbl).

HideScroll(scroll.empl-Tbl, &i,

scroll.Depen-Tbl, &j, scroll.edu-Tbl);

} END-FOR;

Hides all the emp. dependents

} END-FOR;

Education details.

Eq

3

h

g

l

s

01. Depen-Tbl

## ② Referring a Row :

level 1 :

Syntax : Function (Record. <lvl1recname>, <lvl1rowno>);

Eg: HideRow (Record. Empl-Tbl, 1);

↓  
Hides the 1st row of data in Empl-Tbl.

level 2 :

Syntax : Function (Record. <lvl1recname>, <lvl1rowno> | Record. <lvl2recname>, <lvl2rowno>);

Eg: HideRow (Record. Empl-Tbl, 3, Record. Depen-Tbl,

↓  
Hides 2002 dependent details row. 2);

level 3 :

Syntax : Function (Record. <lvl1recname>, <lvl1rowno>, Record. <lvl2recname>, <lvl2rowno> ,

Record. <lvl3recname>, <lvl3rowno>);

Eg: HideRow (Record. Empl-Tbl, 3, Record. Depen-Tbl,

↓ 2, Record. Edu-Tbl, 2);

↓ Hides M.Tech Education details row

## ③ Referring a Field :

level 1 :

Syntax : Function (Record. <lvl1recname>, <N11rowno>, <lvl1recname. <fldname>>);

Eg: &name = Fetchvalue (Record. Empl-Tbl, 3,  
Empl-Tbl. Name);

Level 2:

Syntax: Function (Record. <Lvl1recname> <Lvl1rowno>  
Record. <Lvl2recname>, <Lvl2rowno>,  
<Lvl2recname>. <fldname>);

Eg: &age = Fetchvalue (Record. Empl-Tbl, 3,  
Record. Depen-Tbl, 2, Depen-Tbl. AGE);  
→ Retrieves the age of 2002 dependent.

Level 3:

Syntax: Function (Record. <Lvl1recname> <Lvl1rowno>  
Record. <Lvl2recname>, <Lvl2rowno>,  
Record. <Lvl3recname>, <Lvl3rowno>,  
<Lvl3recname>. <fldname>);

Eg: &grd = Fetchvalue (Record. Empl-Tbl, 3,  
Record. Depen-Tbl, 2, Record. Edu-Tbl, 2,  
Edu-Tbl. GRADE);  
→ Retrieves the grade of M.Tech you

Object Oriented People Code:

Object: An object is a physically (or)  
conceptually thing that you find in the world.

An object is an instance of the class.

Class: A class is a template (or) blueprint for which an object can be created. Class embody all the features of a particular set of objects.

Objects:

→ Objects have

a) Attributes / properties

b) Methods / Behaviour

Attributes: These are the individual characteristics that differentiate one object from another & determine the appearance, state (or) other qualities of that object.

Behaviour: This is the only way objects can do anything to themselves (or) have anything done to them.

Classes: Some of the imp. classes are

- |                |                |
|----------------|----------------|
| ① Array class  | ⑤ Row class    |
| ② Field class  | ⑥ RowSet class |
| ③ File class   | ⑦ SQL class    |
| ④ Record class |                |

Referring An object:

→ To refer an object we need to

- i) Declare
- ii) Create an instance of the object.

i) Declaration: Used to define object scope & shape.

Memory is allotted when an instance is created.

Syntax:

<scope> <class> <objects>;

Eg: Local Record &rec1, &rec2;

Global Field &fld1;

OF

Sc

-

-

-

Eg

ii) Object Instantiation: we can use people code

built in functions to either create a new object

(or) reference an existing object.

Syntax: <objects> = built-in-function;

Eg: &rec1 = CreateRecord(Record.JOB);

&fld = GetField(JOB.EmpId);

After object has been instantiated we can use the dot notation to access the object methods & properties.

Object properties:

Syntax:

→ To set property

Object.property = expression;

→ To get property

variable = object.property;

Eg: &fld1.Visible = False;

&value = &fld1.value;

Field

Dat

→

Built

→

Field

1) Get

2) GE

3) GCF

4) Set

5) SE

& shape

created.

### Object Methods

Syntax:

→ For Methods returns no value

<Object>. <method>(<arguments>)

→ For methods returns a value

<variable> = <object>. <method>(<arguments>)

Eg: &fld1. Setdefault()

↳ Removes value from particular fld

&valid = &rec1. Insert();

### Field class:

Data Type:

→ Field Objects are declared as type field.

Eg: Local field &fld1;

### Built-in-Function:

→ GetField([recordname.fieldname]);

This creates a ref. to a field object for  
the current context (means fields in comp. buffer)

### Field class Methods

1) GetLongLabel(Label ID)

2) GetRelated(recordname.fieldname)

3) GetShortLabel(Label ID)

4) SearchClear()

5) SetDefault()

## Field class properties

Eg:

1) DataArea collapsed

2) Display Format

To find the display format of fields

3) Display only

- Eg: &fld.Displayonly = True

4) Ischanged

Pragmatically whether the field value changed or not

5) Name

Returns the name of field in uppercase

6) Type

Returns the data type

7) value

Returns the value of that field

8) FieldLength

17) Iskey

9) IsAltKey

18) IsListitem

10) IsAuditField Add/ chg/del

19) IsRequired

11) IsDescKey

20) IsSystem

12) IsDuptKey

21) Is Yes No

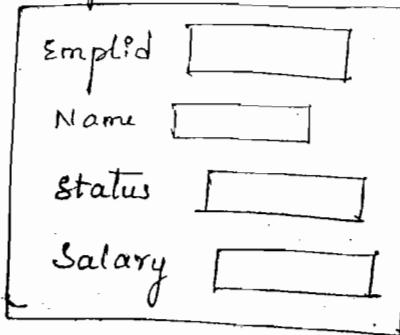
13) Visible

14) IsEditTable

15) IsEditXLat

16) IsINBUF

Eg. Empl-Tbl



- 1) If status = Active then salary is required.
- 2) If status <> Active " " should be hidden.
- 3) Salary field is Display only when seeing existing data.

Write the code in SaveEdit.

1) using procedural coding:

Local character &stat;

Local Number &sal;

&stat = Fetchvalue (Empl-Tbl.status)

&Sal = Fetchvalue (Empl-Tbl.salary)

If &stat = "A" Then AND &Sal = 0 Then

Error ("Salary should be populated");

Using object oriented coding:

status field Field change

Local Field &stat, &sal;

&Stat = GetField (Empl-Tbl. status);

&Sal = GetField (Empl-Tbl.salary);

If &stat.value = "A" Then

&Sal.ISRequired = True ;

End-if ;

2) Using Object oriented coding      status - Fieldchange

Local Field &stat, &sal ;

&stat = GetField (Empl-Tbl. status);

&Sal = GetField (Empl-Tbl. salary);

If &stat <> "A" Then

&Sal.Visible = False ;

End-if .

3) Using object oriented coding .

preBuild, postBuild, RowInit

comp-postBuild .

Local Field &sal ;

&Sal = GetField (Empl-Tbl. salary);

If %.Mode <> "A" Then

&sal.DisplayOnly = True ;

End-if

20/12/08

## Object Oriented people coding

Record class : (I) (R)

→ Record objects are declared as type Record.

Eg: Local Record & myRecord;

Built-in-functions:

1) CreateRecord (Record, recname):

CreateRecord creates a free standing record definition (existing → only structure in component buffer without data populated) & its component set of field objects.

2) GetRecord ([Record, recname]):

GetRecord creates a ref. to a record object for the current context, that is, from the row containing the currently executing program (from comp. buffer).

Record class methods:

1) compareFields (recordObject)

2) copyChangedFieldsTo (recordObject)

3) copyFieldsTo (recordObject)

4) Delete();

Eg: Local Record & Rec

& Rec = CreateRecord (Record, MyRecord);

& Rec.KeyFl = "A"

& Rec.KeyField = "X";

& Rec.Delete();

5) getField

6) Insert()

Eg: Empl-Tbl

Empl	Insert	
Name		101
Salary		A 1000

Local Record & Rec1;

& Rec1 = createRecord (Record: Empl-Tbl);

& Rec1.EmplId = "101";

& Rec1.Name = "A";

& Rec1.Salary = 1000;

& Valid = & Rec1.Insert();

Returns true if insertion is completed.

7) SearchClear()

To clear the search values on search page.

8) setDefault()

To set the values to zero.

Record class properties

1) Field count:

Returns total no. of fields.

2) Field Name: & Rec.GetField(Field.FieldName);

3) Is changed

4) Is

5) N

Row

Row

Eg:

Built

1) Eq

T

Cop

e~

2) C

3) G

4) Get

Row

1) ch

2) D

T

3) I

G

D5

I

- 4) IsDeleted
- 5) Name

### Row Class:

#### Data Type:

Row objects are declared as type Row. For eg.

Eg: local Row &myRow;

#### Built-in-func:

- 1) GetRow();

To obtain a row object, for the current context, that is the row containing the currently executing program.

Eg: &Row = GetRowset().GetRow(Rowid);

- 2) CopyTo(row) — copy a new row of data.
- 3) GetNextEffRow(); — get next future row of current
- 4) GetPriorEffRow(); — get prev. history row " "

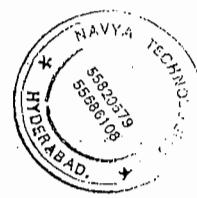
#### Row class properties:

- 1) ChildCount : Returns no. of child rows.
- 2) DeleteEnabled : verify whether the particular row can be deleted (or) not.
- 3) IsChanged :

Returns true if any field value on the primary db record of the row has been changed.

- 4) IsDeleted :

Returns true if the row has been deleted.



- 5) ISNew - Is true if row is a new (inserted) row
- 6) RecordCount - returns no. of records in the row
- 7) RowNumber - returns the row no. of the row
- 8) Visible.

3)  
R)

within the rowset.

R

- 1) C  
2)  
3) F

### Rowset class :

#### Data Type:

→ Rowset objects are declared as type Rowset

Eg: Local Rowset & MyRowset;

#### Built-in functions:

- 1) CreateRowset ({ Record, recname }, & Rowset)

4) F  
5)

[, { Field, fldname, Record, recname }, & Rowset ] ... )

→ The CreateRowset func. creates an unpopulated, standalone rowset

6)

7)

8)

(X) A standalone rowset is a rowset that has the specified structure, but is not tied to any data.

- 2) GetRowset([ scroll, scrollname ])

9)

Use the GetRowset function to get a rowset object based on the current context

10)

11)

12)

13)

14)

15)

Eg: Local Rowset & RS1, & RS2;

& RS1 = GetRowset();

& RS2 = GetRowset(scroll, empl-checklist-item);

1) row

3) GetLevel0()

(R) GetLevel0 creates a Rowset object that corresponds to Level0 of the component buffer.

Rowset class Methods:

- 1) clearDeleteChanges()
- 2) copyToC & copyToRowset [, recordList]
- 3) DeleteRow(n) - Deletes the row in the rowset identified by the parameter.
- 4) Flush() - Remove all rows from the rowset & free its associated buffer.
- 5) FlushRow(n) - To remove a specific row from a rowset & from the comp. buffer.
- 6) HideAllRows()
- 7) Refresh() : Flush + Fill - Reloads the rowset
- 8) ShowAllRows()

Rowset class properties:

- 1) ActiveRowCount
- 2) DataAreaCollapsed
- 3) DBRecordName
- 4) DeleteEnabled
- 5) InsertEnabled
- 6) Level
- 7) ParentRow
- 8) RowCount
- 9) ParentRowset



## SQL class :

### Data type :

→ SQL objects are declared as type SQL.

→ Local SQL & MySQL

### Built-in-functions :

1) CreatesQL([sqlstring[, paramlist]])

2) DeletesQL([SQL.Jsqlname[, dbtype[, effdt]])

3) FetchSQL([SQL.Jsqlname[, dbtype[, effdt]])

used to refer the SQL stmts in App's design

but only select stmts

4) GetSQL(sql, sqlname[, paramlist])

Not Select stmts

5) StoresSQL(sqlstring,[SQL.Jsqlname[, dbtype[, effdt]])

### SQL class Methods :

#### compilation

— checks for syntax

— checks db connectivity

— opens cursor

#### Execution

— executes stmt

— closes cursor

1) c!

2) Ex

3) Fe

SQL

Buc

1) Bul

2) R

3) F

4)

5)

6)

File

Reg

Tran

- 1) close() — Returns true on successful completion.
- 2) Execute(paramlist) — Executes the SQL Stmt of the SQL Object.
- 3) Fetch(paramlist) — Retrieves the next row of data from the Select that is open on the SQL object.

### SQL properties:

Bulk operations on data we use BULKMODE.

#### 1) Bulkmode

To perform Bulk operations

#### 2) Isopen

checks whether the cursor is opened or not.

#### 3) Reusecursor

To reduce the execution time.

→ used when we are executing same Stmt multiple times.

- 4) RowsAffected — How many rows are affected by Stmt.
- 5) Status — whether the Stmt is executed (or) not.
- 6) Value — property is read only.

### File class:

#### Data Type:

→ The file object is an instance of the file class.

A file object is declared using the file data type.

→ Local file & myfile;

### Built-in-functions:

- 1) `fileExists(filename[, pathType])`
  - Absolute (exact path)
  - (\* Test \* .txt) → Relative (default)
- 2) `findFiles(fileSpec-pattern[, pathType])`
- 3) `getFile(filename, mode[, charset][, pathType])`.

Modes:

Append

update

Read

### File class Methods:

- 1) `close()`
- 2) `getPosition()`
- 3) `open(fileSpec, mode[, charset][, pathType])`

Eg: `&myfile.open(&someName, "E", "U");`

- 4) `readLine(string)` — Reads one line of text from the ext. file

Eg: Local file &myfile; Local array of string &myarray;  
Local string &text;

`&myfile = getFile("names.txt", "R");`

`&myarray = createArrayRept(10, 0);`

`while &myfile.readLine(&text);`

`&myarray.push(&text); End-while; &myfile.close();`

5) C

6) C

w

Thi

4) W

E

HC

File

1) I

2) N

3) R

4) S

5) T

6) C

### 5) Setposition(position)

Eg: &myfile : getFile(&somename, "0");

If &myfile.IsoOpen Then

(\* Retrieve the value of the last saved  
checkpoint, &lastpos\*)

&myfile.setposition(&lastpos);

while &myfile.ReadLine(&somestring)

End-While;

&myfile.Close(); End-If;

### 6) writeln(string)

↳ write a string + Enterkey

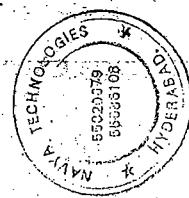
writes the particular line & the cursor waits at  
the new line.

### 7) writeln(string)

↳ writes the particular line & cursor waits at  
the end of line only.

### File class properties

- 1) IsoOpen
- 2) Name
- 3) RecTerminator
- 4) IsError
- 5) IsNewField
- 6) CurrentRecord



2nd year

### Array class :

Array is a variable used to store multiple values of same datatype.

### Data Type

→ Arrays are declared by using the Array type name.

Eg: Local Array of Number & myArray;

Local Array & ArrayAny;

### Built-in-functions :

→ creatArray (paramlist) : creat Array constructs an array & returns a ref. to it.

Eg: &AN = creatArray (6, &AAN [1]);

→ creatArrayAny ([paramlist])

→ creatArrayRept (val, count)

→ split (string, separator)

### Array class Methods :

→ clone() - returns a ref. to a new array, which is a copy of the given array.

→ Find (value)

Eg: &AX      | 1 2 3 4 5 6  
           | A | B | C | D | E | C

&AT. Find ("B")

It returns the index no.

If repetitions Then it will return the index of first occurrence.

→ Get(index)

→ Next(& index)

Eg: Next(i) → (i)+1 = ②

It returns the value in the next index.

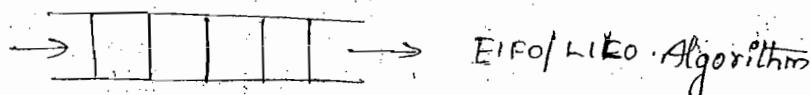
→ Reverse() — used to reverse the indexes of an array.

→ Sort([order]) — used to sort data in a array.

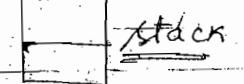
→ Subarray(start, length)

— used to create a array from an bigger array.

### Queue



FIFO/LIFO Algorithm.



Array is nothing but a stack structure.

→ pop() — To pick out the last index value.

push() → To insert value at last index.

push(paramlist)

Array class properties :

Len : returns highest index value

Dimension : returns the dimension of array.

Single dimensional, Two dimensional.

F  
)

E  
f

Traversing a Rowset: 10 comp Etc

(F) (R)

101	A	1000
102	B	2000
103	C	3000
1001	M	214
1002	N	224
2001	X	414
2002	Y	454
SSC	800	B
Degree	700	C
B-Tech	850	A
M-Tech	950	B

err

R

E  
f

Referring a Record:

Refer a level 1 record:

To  
It

① Hide all the emp1-Tbl details.

Local Rowset & RST;

Points to 12 rows of  
data

& RST = GetLevel0() . GetRow(0) . GetRowset(

Record-Emp1-Tbl);

& RST . HideAllRows();

Get the level0 row.

Re  
f

he

E  
f

it

it

### Refer a levels record :

Eg: Local Rowset &RS2;

&RS2 = GetLevel0().GetRow(1);

GetRowset(Record.Empl-Tbl).GetRow(3);

GetRowset(Record.Depen-Tbl);

&RS2.HideAllRows();

Hiding the rows of dependent details of 103 employee.

### Refer a levels record :

Eg: Local Rowset &RS3;

&RS3 = GetLevel0().GetRow(1);

GetRowset(Record.Empl-Tbl).GetRow(1);

GetRowset(Record.Depen-Tbl).GetRow(1);

GetRowset(Record.Edu-Tbl)

&RS3.HideAllRows();

To Hides the rows Tech, Ssc & Degree.

### Referring a Row

#### level1:

Eg: Local Row &Row1;

&Row1 = GetLevel0().GetRow(1);

GetRowset(Record.Empl-Tbl).GetRow(2);

&Row1.Invisible = True; Hides 102 employee row

level 2:

local Row & Row2;

&Row2 = GetLevel0(), GetRow(1).

GetRowset(Record.Empl-Tbl).GetRow(1).

GetRowset(Record.Depen-Tbl).GetRow(1).

&Row2.Invisible = True;

Hides 1001 row.

level 3:

local Row & Row3;

&Row3 = GetLevel0(), GetRow(1).

GetRowset(Record.Empl-Tbl).GetRow(1).

GetRowset(Record.Depen-Tbl).GetRow(1).

GetRowset(Record.Edu-Tbl).GetRow(2)

&Row3.Invisible = True;

Hides Degree row.

(or)

&Row3 = GetLevel0(1).GetRowSet(Record.Empl-Tbl)(1)

GetRowset(Record.Depen-Tbl)(1).

GetRowset(Record.Edu-Tbl)(2)

## Referring a Field:

### Level 1:

```
Local Field &fld1;  
&fld1 = GetLevel0(); GetRow(1);  
GetRowset (Record: Empl-Tbl). GetRow(2);  
GetRecord (Record: Empl-Tbl);  
GetField (Empl-Tbl • Salary);  
&fld1. visible = False;
```

Hides 1002 → Salary details

### Level 2:

```
Local Field &fld2;  
&fld2 = GetLevel0(); GetRow(1);  
GetRowset (Record: Empl-Tbl). GetRow(1);  
GetRowset (Record: Depen-Tbl). GetRow(2);  
GetRecord (Record: Depen-Tbl);  
GetField (Depen-Tbl • Age);  
&fld2. visible = False;
```

Hides 1002 → Age details

### Level 3:

```
Local Field &fld3;  
&fld3 = GetLevel0(); GetRow(1);  
GetRowset (Record: Empl-Tbl). GetRow(1);
```

GetRowset(Record, Depen-Tbl), GetRow(1) •

(3) q

GetRowset(Record, Edu-Tbl), GetRow(1) •

GetRecord(Record, Edu-Tbl) •

GetField(Edu-Tbl, grade);

(4) e

&fld3.Visible = false;

(5) f

Hides the <sup>SSC</sup> BrTech row grade.

(6) n

### Debugging:

(7) e

To find soft errors (or) programming errors,

(8) co

Performed in ways:

① Using Think-Time functions (Win message) (R)

② Using peoplecode Debugger. (I)

To use peoplecode Debugger.

① Create 3-Tier Conn.

↳ performed using config manager.

② Enter Debug mode

APP Designer — open the code for Debugging.

Menu → Debug

↳ Enter Debug mode

(7)

### ③ Select Breakpoints

Menu → Debug

↳ Toggle break at cursor

(place cursor where we want break)

### ④ Execute program.

pt & click as above

### ⑤ Select the variable to be viewed.

### ⑥ Move to next breakpoint (F5)

### ⑦ Exit Debug mode.

### ① Config. mgr.

Startup Tab.

Db Type : App' Server

App' Server Name :

Go To Trace Tab.

provide Local perf & SRV Listener port

ASK sys. admin.

↵

### ⑧ To see the values in variables

Debug → View Local Variables

[ " → " global variables ]

[ " → " comp. variables ]

### ⑨ To exit

Debug → Exit Debug Mode.

we cannot correct the code in Debug. So exit the Debug & correct the code.

6<sup>th</sup> chapter - important

(in Developer's guide).

classes → API reference.

Edit → Find In...

(ActiveRow count)

}

To know Syntaxes.

Re.  
||

→  
→

←

→

←



### Record class:

- A record object, instantiated from the Record class, is a single instance of a data within a row and is based on a record definition.
- A record object consists of 1 to 'n' fields.
- Commonly used method for rec. class - copyFieldsTo  
" " properties " " " → Name,  
Ischanged, Fieldcount.

### Shortcut considerations:

Record.recname.property  
(or)

Record.recname.method (...)

is converted to an object equivalent to

GetRecord (Record.recname)

Eg: &Row = Record.Empl-checklist-Item.parentrow;  
&Row = getRecord (Record.Empl-checklist-Item),

→ Default method for the Record class is 'GetField'.  
Parent row;

Eg: &myRec.Emplid.Enabled = False;  
(or)

&myRec.GetField (Field.Emplid).Enabled = False;

This means you can access a field by just specifying the field name, instead of using 'GetField'.

→ Data Type of record object

These can be declared as type 'Record'.

Eg: Local Record &myRec;

→ Scope of a Record Object

A record can only be instantiated from people code.

→ Built-in-functions

① GetRecord:

If a record is instantiated using GetRecord (either the function or the method), the record object that is instantiated references the record from the current row in the data buffer, & its associated data.

Syntax:

GetRecord([Record.recname])

The full code

&Rec = GetRecord(); is equivalent to

&Rec = GetRow().GetRecord(Record.recname);  
(or)

&Rec = GetRow().recname;

→ We cannot use this func. In peoplecode prgs on events associated with high-level objects like Pages (Activate event) (or) comp. (comp-events)

+ specifying

- With no parameters, this func. returns a record object for the current context (the record containing the program that is running)
- If a parameter is given, Record.recordname must specify a record in the current row.

Eg: Level 2 Rowset (scroll) has 2 records :

Empl-chklist-Itm (the primary rec.) & CHKLIST-ITM-TBL

If the code is running from a field on the Empl-chklist-Itm record, the foll. returns a ref to that record :

&Rec = GetRecord(); /\* Returns primary record \*/

The foll. returns the other rec. in the current row.

&Rec1 = GetRecord(Record.chklist-Itm-Tbl);

→ GetRecord returns a record object.

## ② CreateRecord :

If a record object is instantiated using this func., the record object that's instantiated is a free-standing record def. with its comp. set of field objects in the data buffer. The fields created by this func. are initialized to null values, i.e., they donot contain any data.

→ You can select into this record object using the  
SelectByKey method (or) SQLExec.

Eg:

Syntax:

CreateRecord(Record, recname)

The specified record must have been defined prev., ie,  
it must have a record def. If you are calling  
this func, from people code associated with a page,  
the record does not have to be included on the  
current page.

b)

→ This func. returns a record object that references  
a new record buffer & set of fields.

→ Record class Methods:

a) CompareFields:

Syntax: CompareFields(recordObject)

Compares all like-named fields of the rec-  
object executing the method with the specified  
record object.

The specified record object does not have to  
refer to the same record as the record object  
executing the method.

c)

→ Returns Boolean value ; True if all like-named  
fields have the same value.

Eg: &Rec = GetRecord(Record, OP-Meth-vw);  
&Rec2 = GetRecord(Record, OPC-Meth);  
If &Rec2. CompareFields(&Rec) Then  
    WinMessage("All linked named fields have the  
                        Same value");  
End-If;

### b) CopyChangedFieldsTo:

Syntax: copyChangedFieldsTo(recordObject)

Copies all linked-named fields values that have changed from the rec-object executing the method.

→ This method works only with db records. The comp. processor doesn't track the contents of work records, so there is no changed value to use for copying changed fields.

### c) CopyFieldsTo:

Syntax: copyFieldsTo(recordObject [, DonotCopyUnusedInSource [, DonotCopyUnusedInDestination]])

→ Copies all field values.

→ To restrict the copy to fields that have been marked as unused (set using SetDBFieldNotUsed func.) you can specify either DonotCopyUnusedInSource (or) DonotCopyUnusedInDestination.

→ If you are copying to a derived work record, Ischanged flag for the record is not set.

→ copying fields to a db record does set the Ischanged flag to True.

#### d) DBPatternMatch:

Syntax: DBPatternMatch(value, pattern, case-sensitive)

→ To match string in value to the given pattern.

Value — specify the string to be searched.

Pattern = " " — pattern " " used when searching.

case-sensitive — Takes Boolean value

True — Search is case sensitive.

False — " " not "

→ Returns a Boolean value. True if string matches the pattern.

→ wild characters used are '%' & '-'.

'%' — All characters

'-' — Single character.

eg: M%. — string value starts with M

DATA — finds DATE or DATA

#### e) Delete:

Syntax: Delete()

→ This method uses the key fields of the record & their values to build & execute a Delete SQL Stmt. which deletes the record (row of data) from SQL data table.

+ the

→ This method is like DeleteRow Rowset class method, initially marks the record (or) row as needing to be deleted. At same time the row is actually deleted from the db & cleared from buffer.

→ B'coz this method results in db change, it can be issued only in the foll. events.

i) Saveprechange ii) Workflow iii) Savepostchange.

→ Returns true on successful completion.

f) Executedits:

Syntax: Executedits ([editlevel])

→ Executes the std. system edits on every field in the record.

g) GetField:

Syntax: GetField ({n; Field,fieldname})

→ Instantiates a field obj. for the specified fld. associated with the record.

→ Default method for rec. obj.

→ Returns a Field object.

h) Insert:

Syntax: Insert()

→ Uses the fieldnames of rec. & their values to build & execute an Insert SQL stmt which adds the given record to the SQL table.

→ Can be issued in foll. events

- ① Save prechange
- ② Save postchange

(iii) Workflow

→ Returns True on successful completion.

#### ④ Save:

Syntax: Save([copyToOriginal])

This save method saves the record to the db in steps

→ check to see that it is safe to save this record to the db.

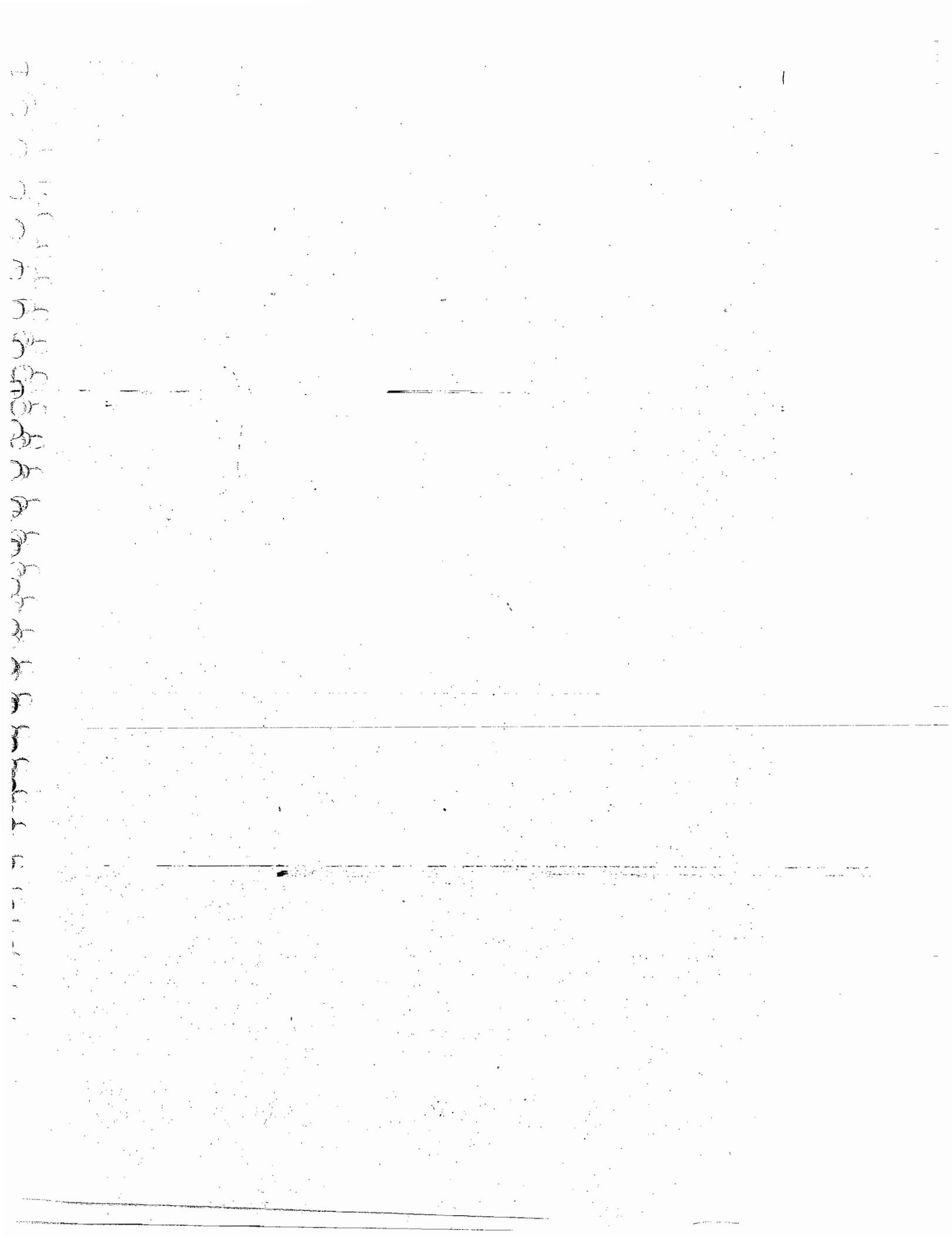
→ either insert this record (or) update an existing record.

→ parameter takes a Boolean value

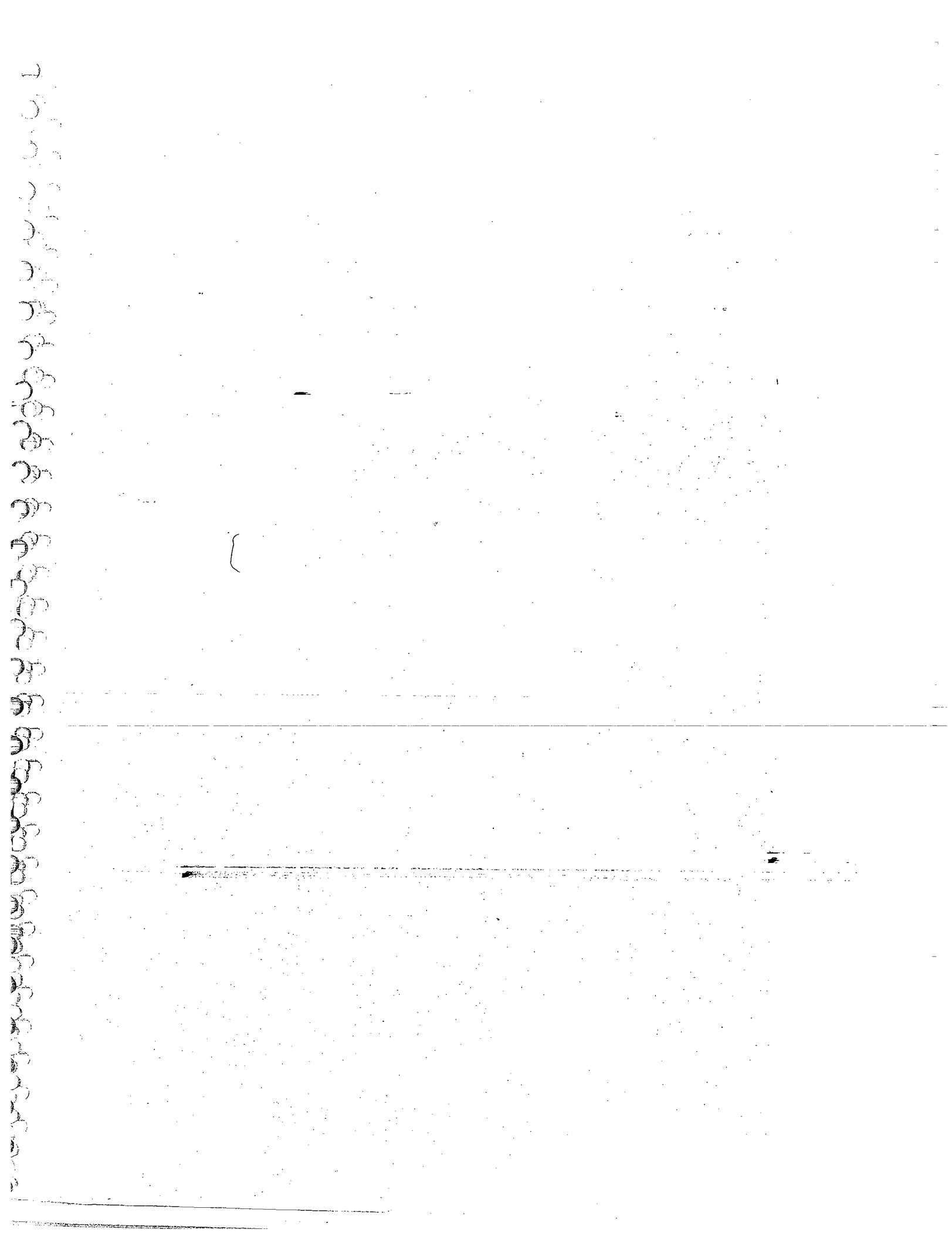
True: The sys. copies the current changed buffers for the rec. into original buffers for rec.

False: buffers are not copied. (Default)

→ Returns true on successful completion.



لهم إني أنت عدو الكاذب والظالم



Constituted and substantiated and made for the benefit of the poor

1860

16/11/08

## Application Engine

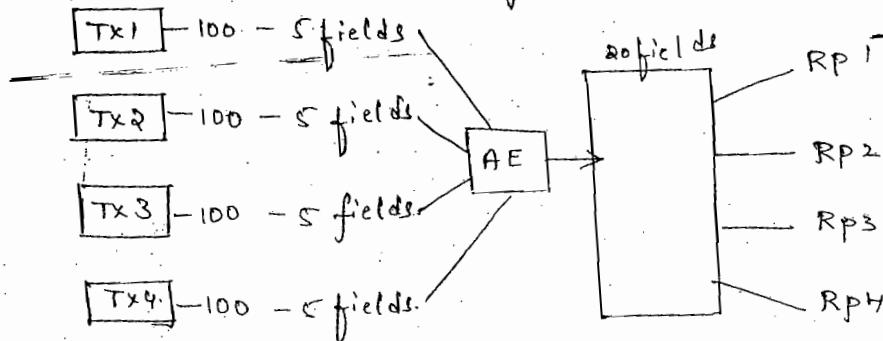
124



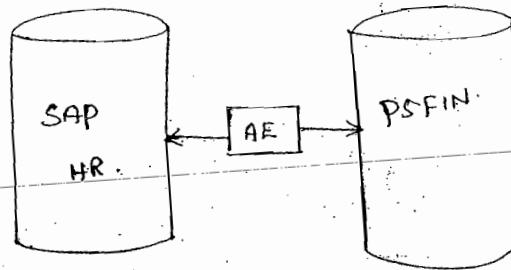
- This is a development tool
- This is a proprietary tool

### Advantages:

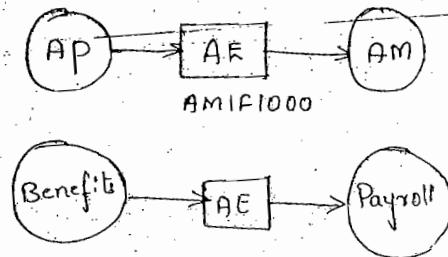
- 1) Used to perform Background SQL processing
- 2) Used to populate reporting tables.



- 3) Used to move data from one database to other.

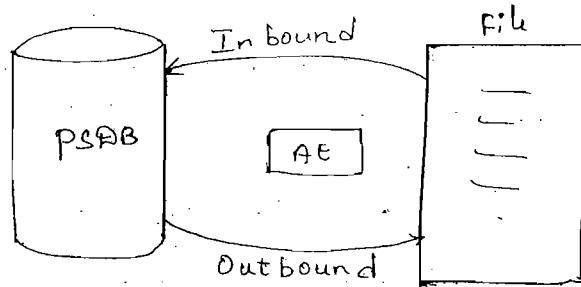


- 4) Used to move data from one module to other.



The process of moving data from AP to AM  
using AE ie called "transaction loader" (AMIF1000)

- 5.) Used to perform file integration.



- 6.) Used in integration tools like file layout, integration broker, component interface etc.

- 7.) Used to perform Bulk processing.

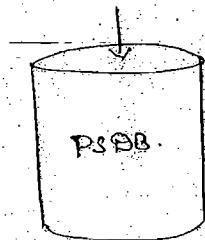
↓  
Operating on huge amount of data.

- 8.) Used to perform parallel processing.

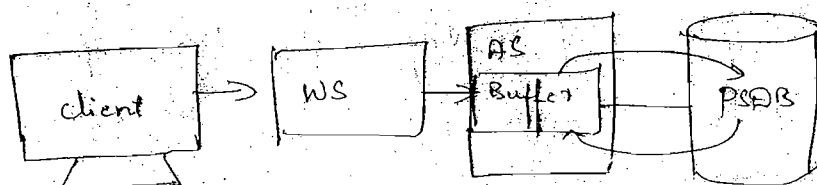
↓  
More than one user can run one prgm at the same time.

- 9.) Used to perform Set processing.

↓  
Executing prgm directly at the db level.



Normal processing.



- 10.) C

g.

Imp

1.) N

→  
the  
→

2.) Q

→

3.) S

→

The

4.) A

→

5.) S

→

The

and

the

the

the

10) Consists of restart facility.

125

↓  
Starting the prgm at the same place where it  
got stopped / aborted.

### Important Elements in AE:

1) Name:

→ This is used to uniquely identify one prgm from  
the other.

→ Every AE prgm will have a name.

2) Section:

→ collection of steps

3) Step:

→ collection of Actions.  
(or)

→ The smallest unit of work in an AE prgm  
that can be committed.

4) Action:

→ Place where source code is written.

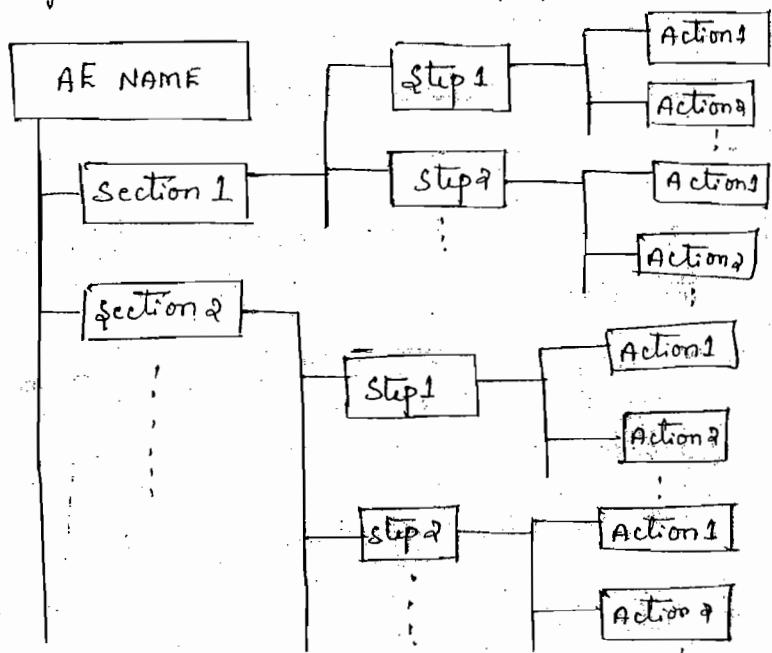
→ We can write SQL & people code statements

5) State Record:

→ This is used to pass values b/w sections  
and steps.

## Program structure:

At a high level how does an AE prgm looks like.



Section: Main section is a place where prgm execution starts.

- Collection of steps.
- we can have n number of sections
- Every section will have a section name
- No 2 Sections in a given AE prgm should have same name.

→ Every AE prgm must compulsory have a section named as 'Main'

Step: → collection of Actions

→ we can have n number of steps in an AE prgm but max 999 in a given section

→ Every step will have a step name

- ;  
→ S.  
→ \$.  
→ &  
→ @.  
→ R.  
  
Actions:  
→ P  
→ SA  
→ C  
→ br  
→ N  
→ F  
→ E  
→ L  
→ D  
→ 2  
→ 3  
→ 4  
→ 5  
→ 6  
→ 7  
→ 8  
→ 9

→ In a given section, no 2 steps should have same name.

→ Smallest unit of work that can be committed.

→ same step & co

→ Diff. sections can have same step name in AF prgm.

Action:

→ place where source code can be written.

→ SQL & people code statements

→ can have n number in an AF prgm/section, but maximum 8 actions in a step.

→ No 2 actions in a given step can have same name.

→ Every Action will have a Action Name.

→ List of Actions provided by ps system are

- 1) Do when
  - 2) Do while
  - 3) Do select
  - 4) Do until
  - 5) People code
  - 6) SQL
  - 7) Call section
  - 8) Log message
  - 9) XSLT
- called as  
Do Actions

AF

23/11/08

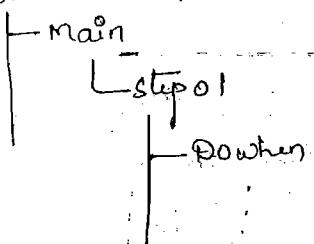
→ In all Do actions we can write select stmt.

### Do When:

→ Used to implement 'If' condition.

→ we can write select stmt.

→ AF



If the select stmt retrieves atleast one row of data then the sys. assumes the cond. is satisfied & executes the below list of actions, if none of the rows are selected then the sys. will not execute the below list of actions & moves to the next step or section.

### Do While: (matters about the count)

→ we can write select stmt.

→ Used to implement 'while' looping.

→ If the select stmt retrieves 'n' rows of data then 'n' no. of times the below actions get executed.

→ If the select stmt retrieves '0'(zero) rows of data

Then

### Do Until:

→ we

→ Use

→ If

but

for

exec

### Do Until:

→ we

→ Use

→ If

(wt)

ex

→ If

then

exec

### SQL:

→ Use

Call SQL

→ Then

Same

then no. of the below actions get executed.

12

DoSelect: (Matters about the count & the values

- we can write select stmt.
- Used to implement 'For' looping.
- If the select stmt retrieves 'n' no. of rows then below actions get executed 'n' no. of times only for the selected values the below actions get executed.

DoUntil:

- we can write select stmt
- Used to implement 'Do while' looping.
- If the select stmt is retrieving 'n' no. of rows (where  $n > 0$ ) then the below actions get executed 'n' no. of times.
- If the select stmt is retrieving zero rows then atleast once the below actions get executed.

SQL:

- Used for writing SQL DML, DDL & DCL stmts.

Call Section:

- This is used to call a section which is in the same prgm (or) call sections from the other prgm.

### People Code :

→ Used to write any people code stmts in an AE prgm.

(1) → The event used in AE prgm is 'ON EXECUTE'.

### Log Message :

→ This is used to write msgs to msg log (or) log file.

### XSLT :

→ This is used to write any kind of coding related to integration broker tool.

### Action Execution Sequence : (I)

1. Do when.
2. Do while.
3. Do select.
4. People code!
5. SQL/call section → These are called mutual exclusive actions.
6. Log message
7. Do UNTIL
8. XSLT.

### Mutually Exclusive action : (I)

→ SQL & call section are called as mutually exclusive actions i.e. we can either use SQL action

COR:

→ The  
(or)

→ F

### Stat :

→ Used  
→ Use a

To the

→ This

in

→ A

(or)

(or) call section action in a given step. 128

In an

→ These are mutually exclusive to avoid deadlock  
(or) commit problems & data integrity.

IN EXECUTE

→ Main

step 01

SQL

update Empl-Tbl

set sal = sal + 100  
where empid = 102

call action

section 01 - same

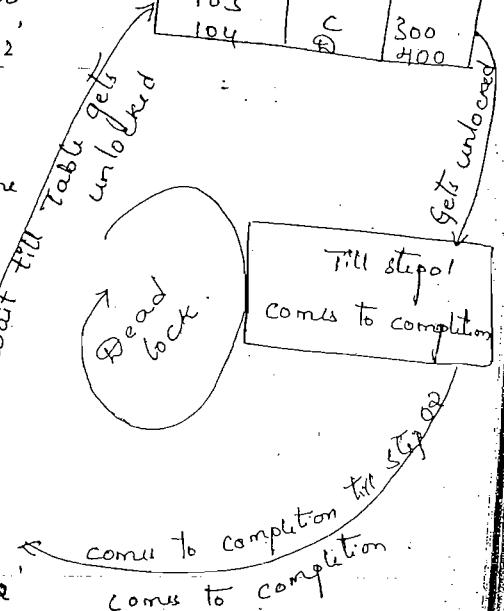
section 01

step 02

SQL

update Empl-Tbl

set sal = sal + 100  
where empid = 102



### Stat Record : I

→ Used to pass values bet<sup>n</sup> sections & steps.

→ Used to pass I/P parameters (runtime parameters) to the AE prgm.

→ This is required whenever a Do Action is used in an AE prgm.

→ A stat record can be a Derived / work record  
(or) SQL Table.

## Conditions to create state Record :

1) Record type should be

a) SQL Table:

Used when restart facility is required for  
AE prgm.

b) Derived / work Record:

Used when restart facility not required.

Synta

% SE

SE

[ h

[ ]

[ ]

Eg:

2) "PROCESS-INSTANCE" must be high level key field.

3) Name must be posto fixed by "-AET".

→ At a given point of time the state record can  
store only a single row of data for the  
provided process-instance.

## Commands to operate state Record:

1) % SELECT { These are used to insert data into

2) % SELECTINIT } state record.

3) % BIND — Used to retrieve data from a  
state record

### % SELECT:

→ Used to insert data into staterecord.

→ Can be used in Do actions & SQL action.

Synta

%

SE

[ h

[ ]

[ ]

Eg:

1

2

3

Syntax:

%SELECT (<Staterecord field list>  
 SELECT <field List> From <Table>  
 [Where condition]  
 [order By]  
 [Group By [Having]]

Eq:

%Select (Emplid)  
 Select Emplid From ps-Empl-Tbl  
 Where Emplid = '102'.

%SELECTINIT :

- Used to insert data into staterecord
- can be used in 'Do' Actions & SQL Actions

Syntax:

%SELECTINIT (<Staterecord field List>  
 SELECT <field List> From <Table>  
 [Where condition]  
 [order By]  
 [Group By [Having]]

Eq:

%Selectinit (Emplid)  
 Select Emplid From ps-Empl-Tbl  
 Where Emplid = '102'

- If the select stmt is retrieving data there is no diff. b/w %select & %selectinit.
- If the select stmt is retrieving zero rows In case of %select the prev. value in the state record will remain unchanged In case of %selectinit the prev. value in the state record will be initialized to 'Null'.

AF

1) AF

2) AF

3) AF

4) AF

To cui

File

To inc

dir

### %BIND:

- used to retrieve data from staterecord.
- can be used in 'Do' actions & SQL actions.

#### Syntax:

%BIND (<staterecord field Name>)

- We can have 'n' no. of state records to a single app engine prgm out of which we can only have one default state record & 'n-1' non-default state records.

To pi

dir

#### Default State Record:

- " " fields can be directly referred

#### Non-Default State Record:

- " " fields are referred by Record Name . field Name

## AE Properties:

- 1) AE program properties
- 2) AE section "
- 3) AE step "
- 4) AE Action "

To create AE prgm

File → New

↳ Select App' Engine prgm. (opens AE prgm window)

To insert section

right click on section → Insert section  
(or)

" " Step → "

(or)

ctrl+1: (or) Insert → Section

To insert step

right click on main section → Insert step/Action  
(or)

Step → "

(or)

ctrl+2

(or)

Insert → step/Action

To Insert Action.

Rt. click on Action → Insert Action

(or)

" Step → "

(or)

Ctrl + 3

(or)

Insert → Action.

Ovr.

has

in

in

Stat

→ me

Qu

in

in

At

in

in

in

in

in

in

in

in

Eg:

File → New → AF prgm.

Insert all actions in step 01.

Save B2H-AF-SEQ

System aligns the actions in the execution order.

Properties → click on icon

(or)

Alt + ↓

General Tab

Description: Short Desc. of prgm

Comments:

long Description

Company commenting status

Owner ID :

Restricts def. to a spec. module.

Last updated :

Date / Time :

By user :

Static Records Tab

→ we can create AE prgm without static record.

Qualify Search

Selected

HR%..AET

Get List

Add

we can add in no. of search records.

Default Static Record.

Advanced Tab

Disable Restart

Select it, it will disable the restart facility.

Application Library

Consists of sections that can be reused in multiple AE prgms.

→ No AE prgm can be created without a Main section but we can create an App' library without a Main section.

### Batch Only

4) 4

To run an AE prgm we have 2 methods

a) Online

On

b) Batch only

Eg:

If we  Batch only we can only execute  
this AE prgm in Batch only method.

We

99%

If Batch only is not selected we can execute  
this AE prgm in Online & Batch only methods.

### Message Set

program Type



1) Standard Only :

These are the normal AE programs what  
we create.

2) Upgrade Only :

These are the AE prgms which are used  
to execute upgrade the version (of people code)

(or) App'

3) Import Only :

These are used for AE prgms which are used  
for Inbound Integration.

Temp

Tempo

→ US

→ T

A-

→ A

→ V

a

2)

3)

4)

5)

6)

7)

8)

#### 4) Daemon Only :

These are the prgms which are executed only once & these prgms will keep on executing

Eg:

Delivered app' prgm PSDAEMON

99% we don't create Daemon only AE prgms.

Temp Table Tab.

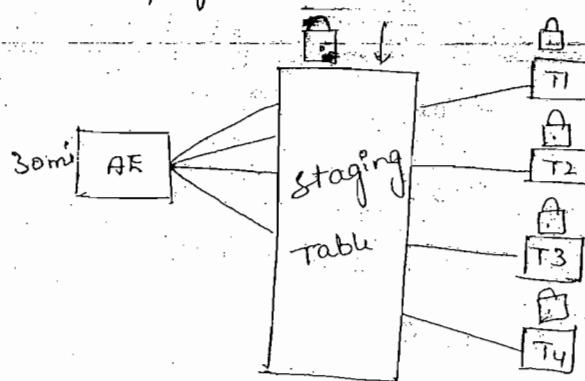
Temporary Tables:

- Used to increase the performance of AE prgm.
- These are also called as staging table for AE prgm.

→ Adv. of staging tables:

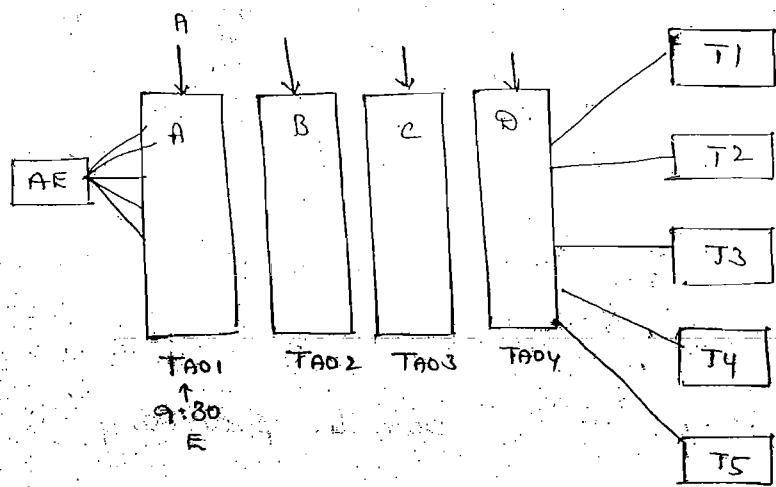
① Used to avoid locking of actual tables for a longer period of time.

② Used to implement parallel processing in AE prgm.



User	Request Time	Starts execution	Completion time	Waiting time
A	9.00	9.00	9.30	0
B	9.05	9.30	10.00	25min
C	9.10	10.00	10.30	30min
D	9.15	10.30	11.00	1.15min
E	9.20	11.00	11.30	1.40min.

→ Serial processing — After the completion of one user processing, then only next user req. processes.



User	Req-Time	starts exec.	Completion time	Waiting time
A	9:00	9:00	9:30	0
B	9:05	9:05	9:35	0
C	9:10	9:10	9:40	0
D	9:15	9:15	9:45	0
E	9:20	9:30	10:00	10min.

→ Parallel processing.

Cond.

- 1) The
- 2) " PE
- 3) Narr

Qua

[H]

Jr

sp

at

For

be

Re

T

L

C

C

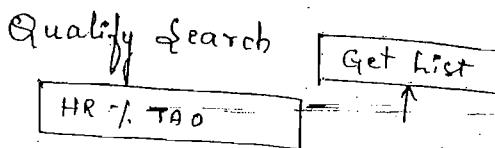
C

C

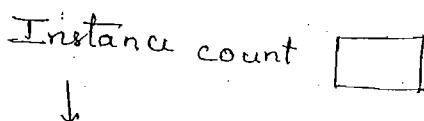
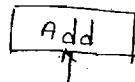
~~Waiting time~~ = Request Time - Start of execution time.

Conditions to create Temporary Tables:

- 1) The record type must be temporary tabl.
- 2) "PROCESS-INSTANCE" must be high level key field.
- 3) Name must be postfixed by -TAO (or) -Tmp



Selected



Specifies how many copies need to be created at runtime

For better performance the instance count should be 4.

Runtime

If non-shared Tables cannot be assigned

The copies are not created

O continue → do serial processing

O Abort

## Section Properties:

Section 1. Section 1 description.



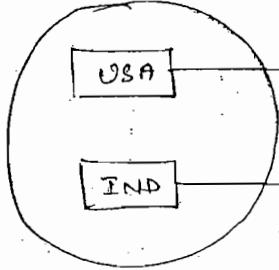
Section Name

Market



GBL

Used to write country specific coding



Tax

0 to ∞ - 30%

0 to 1Lakh - 20%

1L to 2L - 10%

We can have same section name with diff

Market names : market { USA — Sec1  
Names { IND — Sec1

platform

default

Effective Date

Used to maintain code date wise as well as  
keep our data as it is.

IND - 0-1L	0%
1-2L	10%
2L-3L	20%
3L-4L	30%

as per 2007.

IND - 0-1.2L	0%
1.2-2L	10%
2L-3L	20%
3L-4L	30%

as per 2008

Eff

Spec

exe

secti

Auti

□

Cp

secti

Acti

L

Wc

Stop

Stop

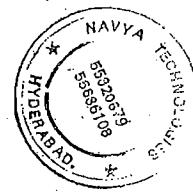
Stop

## Effective status

Specifies whether this section needs to be executed (or) not

Active — Executed

Inactive — Not executed.



## Section Type

prepare only — when operating on Normal TXI

critical update — when operating on critical data

## Auto commit

After step If we deselect this then commit depends upon step properties of this section (or) of other steps sections

## Access

public

we can called from other AE prgms

## Step properties

Step 01. Step 01 description



Step Name Specifies short desc.

Commit After

Default	<input checked="" type="checkbox"/>
---------	-------------------------------------

Late

After step

Supp

Action

Do

Do

If  
exec

If  
rest

yes  
no

at

Bulk

use

Section →	<input checked="" type="checkbox"/> After step	<input type="checkbox"/> After step
Step ↓		
Default	commits after step	Not commit
After step	commits after step	commits after step
Late	Not commit	Not commit

Frequency

Specifies after operating on how many rows system issue a commit to db.

On Error

Abort	<input checked="" type="checkbox"/>	writes to log file & stops execution
Ignore	<input type="checkbox"/>	
SUPPRESS	<input type="checkbox"/>	

ignore - writes error msg to msg log & continues the process

~~Do when~~ — sys ignores the error msg, continues the execution, it will not write msg to log file.

Action

↳ Executed.

Action

↳ Not executed.

### Action Properties :

~~Do when~~

~~Do when desc.~~

↳ Action Name

↳ short descriptions

Reuse statement

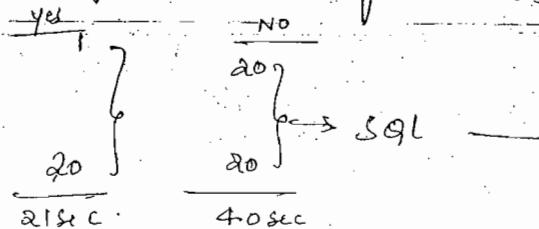
~~Do while~~

~~Do while desc.~~

Reuse statement

If reuse stmt as 'No' every time it compiling & executing.

If reuse " as 'Yes' only once it compiling rest of time it only executes



compile

1sec checks syntax  
create cursor

Execution

1sec execute  
close cursor

Bulk Insert

Used to perform bulk processing

Do & do

Do & do type

Restartable

Select / Fetch

ReSelect

Empl-Tbl.

DeptId	EmplId	Name
10	101	A
10	102	B
10	103	C
20	201	D
20	202	E

X-Y-Z - RET

PI	EmplId	DeptId

ReSelect → selects value only once & place it in  
state record

①. SELECT (DeptId)

Select DeptId From Empl-Tbl  
Where EmplId = '10'

②. Select (EmplId)

Select EmplId From Ps-Empl-Tbl  
Where DeptId = '10'

People code

on Return sys-stops executing prgm

Abort, Break, skip & p

IS

SK

SQL

Rec

Co

SK

Log

Call

SC

L

me

public

prgm

Dyn

This

prgr

Break → sys stops executing this piece of code & goes to next action.

Skip step → sys stops executing "particular step" & goes to next step.

### SQL

NO Rows — Abort.

Section break — jumps from this section to other.

Continue — continues the next step.

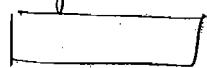
Skip step — skips this step & goes to next.

### Log Message

Message set      NO :      parameters

### Call Section.

Section Name      Program ID



Dynamic

we can see only public sections of the program selected.  
Used for dynamic calling of sections.

### Dynamic Call Section:

This is used to call sections dynamically (or) programmatically at runtime depending upon other values.

## Conditions :

- 1) To dynamically call a section the default state record of the AE prgm must have program\_id & section\_id fields.
- 2) Before coming to call section we need to make sure program\_id & section\_id in the default state record must be populated with prgm name & section name resp.

Eg:

Select \* from PS-NAMES

Need To change name RAMESH,B to Ravi,K where  
Emplid = 0021.

Main

↳ step01

↳ SQL

Update PS-NAMES

Set Name = 'Ravi,K'

where emplid = '0021'.

File → New → Select App engine prgm

Select Action

→ click on Step

↳ Insert Action  
(SQL)

If

wind

pre

l

u

Double click on SQL

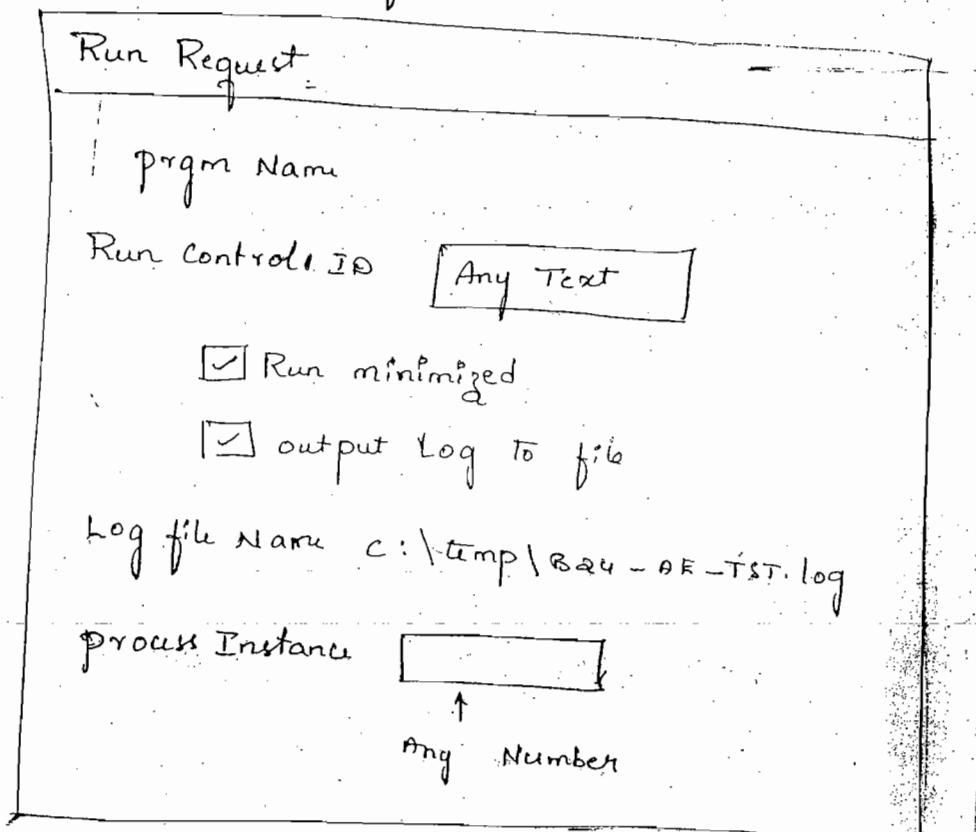
Update PS-Name

Set Name = 'Ravik'

where empid = '0021'

Save B24-AE-TST

click on Run request icon.



If we deselect Run minimized a black color window appears thro' out the execution time.

Properties → Advanced

Disabled Restart

## AE Execution Methods: (I)

### ① Online Execution Methods (2-tier execution)

- a) Using App. Designer → Used for development environment.
- b) Using Command line prompt
- c) Using People code.

### ② Batch only (n-tier execution) — for clients.

- a) Using User defined run control component
- b) Using Request AE component.

① b) → PSAE.exe — Executable file which is used to run a AE prgm.

path of PSAE.exe

<Drive>\pt8.4x\bin\client\winx86\PSAE.exe

For this PSAE.exe to run we need some i/p parameters.

Start → Run → cmd

Go to the path <Drive>:\pt8.4x\bin\client\

winx86

OP

Rnc

PSU

-CSE

-CE

-CA

-CP

-R

-A

-I

-DE

-D

-T

-DE

-TOC

-TOR

-OT

-OF

-FP

Using

→ CC

(I)

open any page in portal & press **ctrl+J** to know the APP Server name.

**psae -cT dbtype** - Eg: ms sql, oracle etc }  
-cs server — App'server Name  
-ca database\_name - Eg: HR, FIN, HRDEV... }  
-co opriD — Eg: ps, vpi...  
-cp opri#pswd? — Eg: ps, vpi... }  
-R run-control\_id — provide Text  
-AI program\_id — AF program Name }  
-I Process\_Instance — Any Number }  
-DEBUG (Y/N)? — Y - Debugging, N - No debugging }  
-DR (Y/N) — Y - No restart, N - restart.  
— Tracevalue Tracevalue }  
-DBFlags flagvalue } Used for tracing.  
-Toolstracesql value? }  
-Toolstracepc value }  
-OT outtype — Eg: file, web etc  
-OF outformat — Eg: PDF, CSV, HTML etc.  
-FP file path — Eg: path of o/p file.

Using people code:

→ callAppEngine is the function used to run  
① AF prgm

Syntax: (I)

V.Sir.

call appengine (<prgmid>, <state record>)

D.C.

Using userdefined run control component:

TEST AF

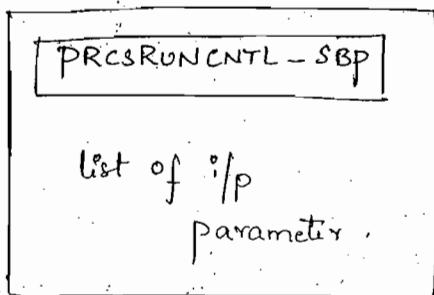
1) Create Required Run control definitions

↓  
prgm.in  
TEST Tier

a) Run control Record

(Same as in SQR)

b) Run control page



c) Run control component

Page: Runcontrol page

Search Record: PSRCSRUN CNTL

Register the component.

d) Process definition creation.

people Tools → process scheduler → processes.

prev.cgi

This is without i/p param.

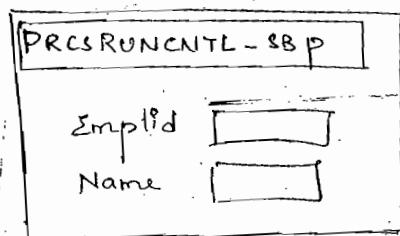
## Using IP parameter AR3TIER - project

- 1) Create Runcontrol record (B24-AE-RUN)

oprid	Run-cntl-id	Emplid	Name
K	K		

- 2) Create runcontrol page  
(B24-AE-RUN-PG)

PS TST 0022 Shiva,A  
VPI TST 0021 xyz,M



- 3) Create run control component

page : B24-AE-RUN-PG

Search Record : PSPRCSRUNCNTL

Go to oprid → Rowinit event

oprid.Rowinit

↳ B24-AE-RUN:oprid.value =

PRCSRUNCNTL.oprid.value;

oprid.RUN-CNTL-ID.Rowinit

↳ B24-AE-RUN.RUN-CNTL-ID.value =

PSPRCSRUNCNTL.RUN-CNTL-ID.value;

Main

↳ step 01

↳ SQL

update ps-Names

set Name = <value>

where Emplid = <value>.

Creation of state record (xyz-AET)

process-instance, Emplid Name

K 0022 shiva, A

Main

↳ step 01

→ ~~SQL~~ when

◦ Select (Emplid, Name)

Select Emplid, Name From ps-BASE-RUN.

where oprid = % operator ID

AND RUN-CNTL-ID = % Run control.

→ SQL

update ps-Names

set Name = % BIND (Name)

where Emplid = % BIND (emplid)

Using Request AF component:

1) Create process Definition.

Go to portal

↳ people Tools → process scheduler

↓  
processes

process def.

↳ component

B24 - AE - RUN - CMPT

AE - request

change the name as Ramesh, B.

→ Navigation

Select, \* from PSPRSMDEFN

where portal-objectname like '%.component'

Name %

Select the last field copy it & paste it  
in portal to know the nav.

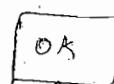
→ people Tools → AE

↳ Request AE

Add a New Value.

User ID : PS

program Name : B24 - AE - TST



process Frequency : Always

## Using Request AE Component : TESTAE1

- 1) Add AE component request.

people tools → process scheduler

↓  
processes

Main

↳ step1

↳ SQL

Add a New value

Process Type: App Engine

Program Name: TestAE1

Add

Go to process Definition option

component

AE-REQUEST

process Groups

All Pages

Save

RunRec

↓  
OPRID

Key { RUN-CNTL-ID

Emplid

Name:

- 2) People Tools → Application Engine

↓  
Request AE

Add a New Value

User ID : PS

Run control ID :

Program Name : [TestAE1]

Add

3)

This open AE Request window

process Frequency :

Market :

Parameters

State Record :

Bind variable Name :

Value :

click on

State Record :

Bind variable Name :

Value :

Then click on

③ Go to people Tools

This open process & scheduler Request

Select the program  TestAE1

4) PeopleTools → Process Scheduler  
↓  
process monitor.

check the status.

For Input parameters prgm running thro' AE request method, there is no need to write the Do when Action, b'coz we are directly giving values to the p stage state record in AE request window.

→ Without AE request method if we are running then write the foll codes. (For input params)

In Rowhen Action

```
%-Select (Emplid, Ename)  
Select Emplid, Name  
From ps_RUNREC  
where OPRID=%operatorid  
AND RUN-CTL-ID = %Runcontrol.
```

SQL-Empl-Tbl  
↓  
Emplid  
Ename  
Basic  
TA  
PA  
HRA

In SQL Action

```
update ps-SQL-Empl-Tbl  
set Ename = %Bind(Ename)  
where Emplid = %Bind(Emplid)}
```

This is required for  
all methods of  
%p parameter case

30/11/08  
Tru

Kn.  
Tear

①

②

③

④

→ Add the static record to the App' Engine  
prgm while doing input parameters.

properties → static Record



Select the Name (XYZ-AET)



Add to the List

30/10/08

### Tracing of AE program:

→ we can know the prgm flow as well as we can  
know where the time taking in our AE prgm.

#### Tracing Methods:

##### ① Using Config. Manager:

→ Used in 2-tier execution of AE, specify the path

→ Not user Dependent & not prgm dependent

##### ② Using App. Server config. file:

→ Used in n-tier & 2 tier execution of AE.

→ Not user Dependent & not prgm dependent.

##### ③ Using Batch Server config. file:

→ Used in n-tier execution of AE

→ Not user dependent & not prgm dependent

##### ④ Using Process Definition:

→ Used in n-tier execution of AE

→ Not user dependent but prgm dependent.

- ② Trace file is located on App' server.
- ③ Trace file is located on batch server
- ④ Trace file is located on Distribution server.

Location on process scheduler.

I

or

②

→ 99% we can't perform tracing using App' server config file & Batch server config. file.

→ Using process Definition Method is mostly used.

① open config manager:

Go to Trace Tab

SQL

SQL — writes all SQL there op's to trace file.

Ded. Temp Table — operations performed on temp table.

Stmt Timings (file)

" (Table) → writes all timings to table

people code Detail Timing

DB optimiser (file) } used to check the conn'g speed

" (Table) } used by DBA

people Tools Trace file   (path)

If we don't specify this, it creates trace file  
on the same loc. of log file.

## (2) App' server config file:

E:\PT8.45\appserv\HR

<Drive>:\pt8.4x\appserv\<Database Name>

\psappsvr.cfg

Just copy the config file & see the statements.

Do not modify this app' server config file.

[Find = traceae]

Specify the bit numbers to perform tracing.

Trace AF = 129

(Step trace + timing trace).

## (3) Batch server config file:

<Drive>:\pt8.4x\appserv\prcs\<DatabaseName>

\psprcs.cfg

copy this file & open with Notepad.

Find = traceae

which is used to start your process scheduler.

## (4) Process Definition:

Open the program what we want to trace in

PT → process & scheduler

↳ processes

↳ open the prgm what we want  
to trace :

Go To override options page

Command line : Append

Parameters :

Specify the commands i.e flags (SIR will send this)

It creates the trace file for the AF flag

Then again come back to here & remove the

flag & set command line to None

Debugging :

→ used for finding soft errors (prgm errors)

Steps to be followed:

- 1) Create 3-tier conn'
- 2) Enter Debug mode
- 3) Run AF prgm in 2-tier
- 4) Select appropriate options & Debug prgm.
- 5) Create 3-tier conn'

using config.mgr

↳ Select App' server  
& give Name

2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  

2) To enter Debug mode

config manager

↳ profile Tab

↳ click on Edit

↳ process & scheduler

↳ select Debug

Debug

3) config.mgt

↳ Trace Tab

↳ Local pSERBGSRV

ASK system ← listener port 9500  
administrator.

3) open prgm

click on Run icon

This opens black window, it will not disappear

until we give the options

Just click give ? & ↲

It provides the list of options

Quit (Q) → Rollback & exits from prgm

Exit (X) → Commits & comes out of prgm

Commit (C) → gives commit to prgm pragmatically

Break (B) → list of steps & issue breakpoint  
where we want to stop ex.

Look (L) → Is a command which is used to see the values in state record.

Modify (M) → Used to change values in state record.

Watch (W) → used to set watch fields. sys will automatically gives break when the values are changed.

Step over (S) → Execute the current step to completion & stop at the next step in current section.

Step into (I) → In the same step, we can issue break for each & every action

Step out of (O) → comes out of action breaks in action level.

Go (G) → move from that particular break point & continue the process.

Run to commit (R) → Resumes execution of your prgm after it has stopped.

## Process Monitor :

**Process List** Tab

**Refresh**

User ID :  Type :  Last  Days \*

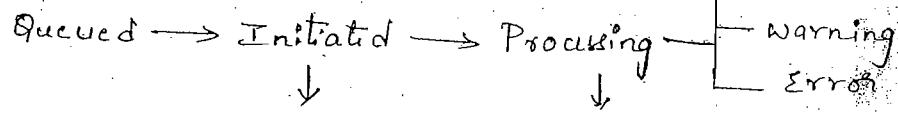
Server :  Name :

Instance :  to

Save on Refresh.

Run status :  Distribution status :

Status :



Req. is taken by whenever server  
server is executing  
the program.

If the process is in Queued state for long time it means  
1) server may be busy  
2) server is not working.

To see whether server is working (or) not

Go to **Server List** Tab.

If status : Running then it is working.

→ Recurrence - Scheduling the program

PT - process scheduler

↳ Recurrences

If we want to run the process for every 2 hrs  
then we use recurrence.

Recurrence Name : B25- Recur

Recurrence Def Tab.

Schedule Next Recurrence when

- Current req. is initiated
- Prior recurrence has completed ..

Rec. pattern :

- Daily
- Every day

Start Req.:

Date : 12/16/2008

End Req.:

Date : [ ]

Time : 9:00:00 AM

Time : [ ]

Repeat:

Every [ ]

For [ ]

- Do not schedule any processes missed from the recurrence pattern.

we have to add this recurrence to the app' that need to be run for every 2 hrs.

Eg: Go to AF program in portal.

↳ Give Recurrence B25- Recur [OK]

Gu

flow

Nav

Ao

T

D

To

R

F

See

sel

Rul

JX

We

He

C

Co

G

the

el

el

el

every 2 hrs Go to process monitor & check

How to create a JOB?

Nav: PT → process scheduler → JOBS

Add a New value.



B25 JOB : Name

↗

JOB is collection of programs (or) processes.

Run mode: serial

Priority: medium.

Select the programs what we want

Select the options for Run always on warning &  
Run always on error.

If we don't select this if any program has  
warning (or) error it does not allow to process  
the 2<sup>nd</sup> program i.e. the programs next to it

comp:

Select a comp. from which we want to run  
the job.

17/12/08

## Server:

WS, AS, Batch server, db server.

Two methods to connect to AS:

- ① Remote Desktop.
- ② citrix



citrix Nflex.

Start → progs → Accessories → communication

↓  
Remote Desktop.

Whenever we click it asks for the IP address of the AS.

↳ connect.

- ② citrix is another s/w which is used to create connection to AS.

## Mailing s/w in company:

① outlook

② Lotus Notes

↳ Microsoft



↳ App' developed by IBM.

③ Snagit

→ used to take screenshots of app'.

④ ultraedit-32

→ similar to Notepad, we can see line no & col no.



→ To unzip a Zip file.

zipgenius

→ s/w provided by OS, we can see the NetMeeting desktop of other user.

→ When we double click of Lotus Notes it asks for password.  
Provide & click on OK.

It takes to foll. page



→ When ever we double click on citrix window

Add ICA NW

Select LAN → Next

Provide As Name in new ICA conn.

Next

It creates separate icon on the desktop.

double click

UserName :

Password :

Logon

OK

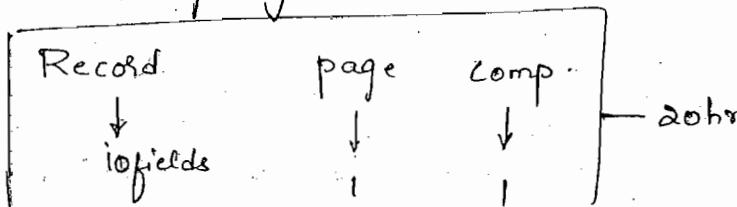
→ Want a page, to enter (or) change emp sal.

1st we estimate ← Time estimation

Consider about ←

- how many resources

- complexity -



First we don't create a table we have to  
prepare a design documentation.

Steps during deve.

Estimation

— Developer / Sr. Developer

Documentation

— Developer

Design Review

— Developer with B.

Development

— Developer

Comments

— Developer

unit Testing

— Developer

Development Review / code Review

— Developer with

Sr. Developer

Migration

— System Administration

Testing

— QA Team

Production

— System Administrator

→ Version control s/w's used for design documents

Star Team, Share point, Stat

- Every company has internal portal which is used to know HR policies, about company etc.
- VoIP - voice oriented Internet protocol.  
↳ leased line for us.  
we can call to us clients to interact with them.
- For chatting within company we have SW in nodes called as Same Time.  
(or) by using NetMeeting.

Srinivasarao.chittineni@capgemini.com  
Srinivasarao@capgemini.com

caesar | mnb@rennertsystems.com

Khadupalle Krishnamoorthy  
Bharadwaj Venkateswaran  
Mridumathi

~~Nagpur~~ Juna Karna

⇒ SQR

⇒ Crystal Reports

⇒ Integration Tools

⇒ File layout

⇒ Tree Manager

⇒ Data Mover

⇒ Nvision

⇒ Finance Functional

⇒ HRMS. Functional

⇒ Work Flow

Page 53

D: Satyam erayana:

(1)

1  
2  
3  
4  
5  
6  
7  
8  
9

1  
2  
3  
4  
5  
6  
7  
8  
9

1  
2  
3  
4  
5  
6  
7  
8  
9

1  
2  
3  
4  
5  
6  
7  
8  
9

1  
2  
3  
4  
5  
6  
7  
8  
9

1  
2  
3  
4  
5  
6  
7  
8  
9

1  
2  
3  
4  
5  
6  
7  
8  
9

## SQR

### [Structured Query Reporting]

- This is a reporting tool.
- This was initially owned by MIT Corporation, now this is owned by Brio Corporation.
- Supports all SQL statements (DML, DDL & DCT).
- Supports all databases and all operating systems.

#### Terminology:

Report: Generates o/p.

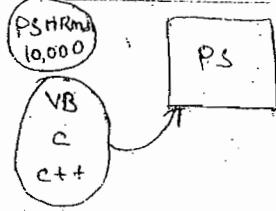
Getting the data from db & displaying it in a particular format.

Process: Does not generate o/p, but goes to backend & perform some manipulations.

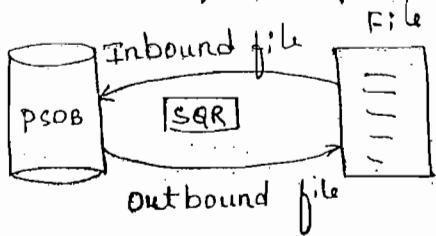
Job: Collection of report / processes.

#### Advantages of SQR:

- used for Reporting.
- used for Back ground SQL processing to update db.
- used to migrate (or) move data from legacy system to people soft system.



→ used to perform file integration



→ used where programming logic is more

→ this can be used both as GUI & CUI.

upto ps 7.5 SQR

Supports GUI thro'  
SQR work bench.

character user if.

But from ps 8.

SQR supports CUI.

### SQR Commands :

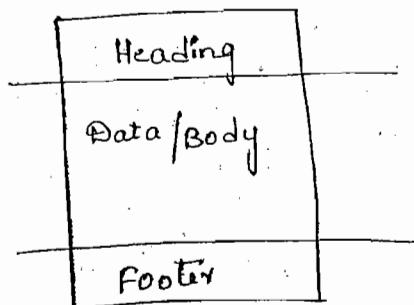
Three types

i) Sections,

ii) Paragraphs

iii) Other commands

O/P of SQR is classified into 3 regions



### Sections:

i) Begin - Setup :

→ used to perform o/p page setup like page size

Fonts format, font size etc.

(2)

→ Maximum we can have only one section

→ Not a mandatory section. (System will provide a default section)

→ Syntax:

1	2	3	4	5	6	...
1	Begin-Setup					
2		Other commands				
3						
4		End Setup				
5						
6						

→ SQR is a space sensitive language, not case sensitive.

### ii) Begin-Heading:

→ Used for heading on each and every o/p page.

→ Max. only one section

→ Not a mandatory section.

→ Syntax:

1	2	3	...
1	Begin-Heading (#)		
2		Print commands	
3			
4	End-Heading		
5			
6			

This number specifies the no. of lines that heading should occupy.

Eg: 3, it reserves first 3 lines for heading.

### iii) Begin-Footing:

- used for footing on each and every o/p page
- Max. only one section.
- Not a mandatory section.

→ Syntax:

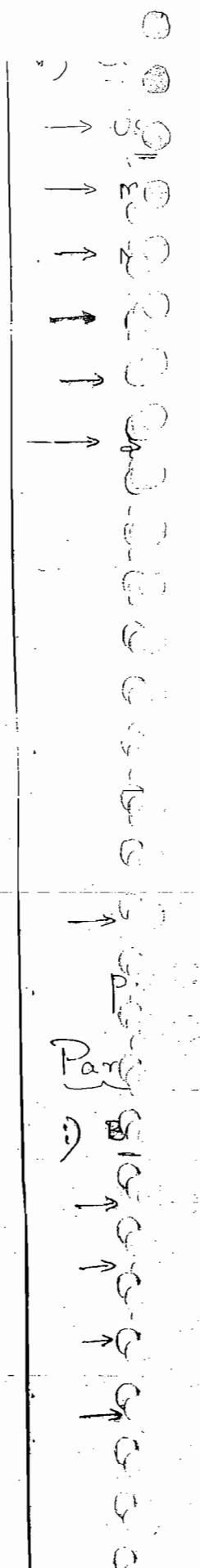
```
1 2 3 4 . . .
1 Begin-Footing #
2 print commands
3
4 End-Footer
```

### iv) Begin-Program:

- used for Data/body region.
- Max. only one section.
- Mandatory section.
- This is a place where prgm execution starts

→ Syntax:

```
1 2 3 4 . . .
1 Begin-Program
2
3 other commands
4
5 End-Program
```



Part

### v) Begin-Procedure

- Used for data/body region.
- Max. we can have 'n' no. of sections.
- Not a mandatory section.
- Good programming practice.
- Reusability & easy maintenance.
- Syntax:

```
1 2 3 4 . . .
2 Begin-procedure <procedure-name>
3
4 other commands
5
6 End-procedure [<procedure-name>]
```

→ we cannot write one procedure in other procedure, we can just call it.

### Paragraphs:

#### i) Begin-Select:

→ Used to select data from database.

→ We can have max. in no. of paragraphs.

→ Not a mandatory paragraph.

→ Syntax:

Begin-Select

Field 1

Field 2

Field 3

:

FROM <Table list>

[WHERE <condition>]

[ORDER BY J]

[GROUP BY [HAVING]]

End-Select

→ US.

↓

↓

↓

↓

↓

### i) Begin-SQL :

→ Used to write DML, DDL, & DCL SQL stmts.

→ Max. n number.

→ Not a mandatory paragraph.

→ Syntax:

Begin-SQL

SQL statement 1;

SQL statement 2;

SQL statement 3;

:

End-SQL

SQR

↓

↓

↓

↓

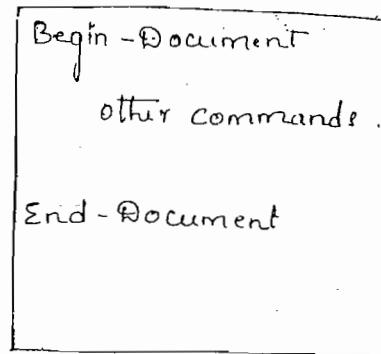
↓

↓

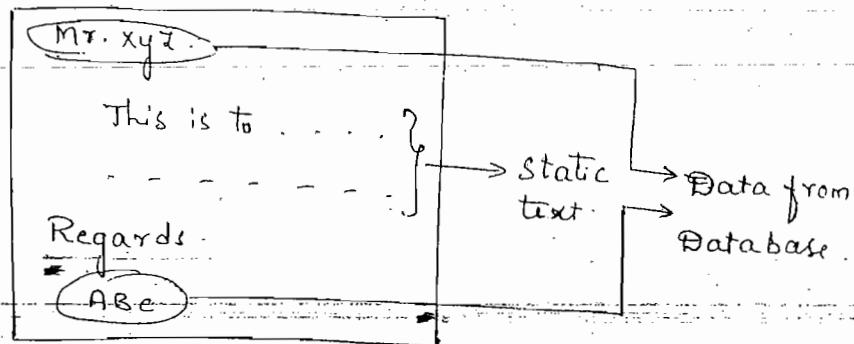
### iii) Begin-Document :

(4)

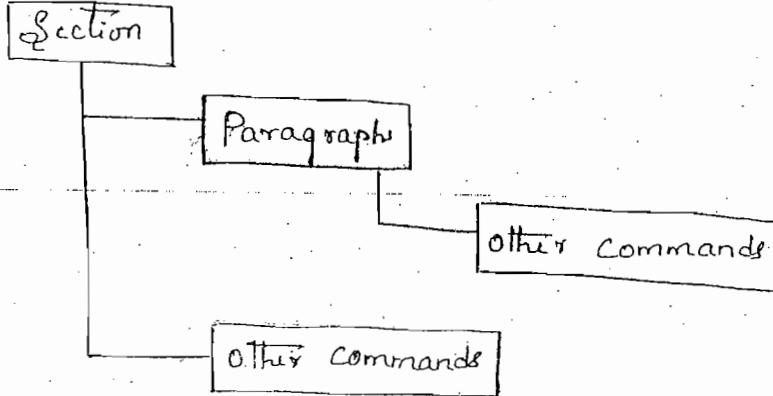
- Used to combine static text with data from database.
- Used in "Letters and Document" type of report.
- Max. only one.
- Not a mandatory paragraph.
- Syntax :-



Eg:



### SQR program structure :



Sections	Paragraphs		
	Begin-Select	Begin-SQL	Begin-Document
Begin-setup	X	✓	X
Begin-Heading	X	X	X
Begin-Footing	X	X	X
Begin-program	✓	✓	✓
Begin-procedure	✓	✓	✓

### PRINT Command:

→ This is a mandatory command for any SQR report

→ Two types

i) Implicit : No print command

ii) Explicit : print command required, used mainly for static text.

→ Formatting options

- Bold

- Center

- Underline

- Box

• Shape → Moving from one-line to another if  
the text is long.

- Wrap → opposite to bold

• Edit — used to print text in specified format

- On-Break

Eg: MM/DD/YYYY

→ ON-BREAK:

- used for redundancy of printing

- Group Headings
- Group Footings → Default option
- Options: change, change/Top-Page, Never, Always  
After, Before, Save.

→ Variable - Temporary memory location to store  
a value.

Field - permanent " " " "

a value.

### i) Implicit print:

Syntax:

<Variable> (<Rowno>, <colno> [, <length>])  
print co-ordinates.

### ii) Explicit print: [<Format>]

Syntax:

print <static Text> (<Rowno>, <colno> [, <length>])  
[<Format>]

### ON-BREAK

Ex:- COUNTRY (1,1) ON-BREAK LEVEL=1  
State (1,10) ON-BREAK LEVEL=2.  
Empid (1,20)

IND	101	NJ	106
	102		107
up	103		
up	104		
USA	NY	105	105

### Syntax:

country (1,1) ON-BREAK

" " ON-BREAK = change/Top-page

" " ON-BREAK = NEVER - never print the value

" " ON-BREAK = ALWAYS

" " ON-BREAK = AFTER = <proc.name>

- Save <variable>

" " ON-BREAK = BEFORE = <proc.name>

" " ON-BREAK = LEVEL = <proc.name>

①

Start of IND Emp

IND AP 101

102

UP 103

END of IND 104

S.of USA 105

USA NY 105

106

②

USA NJ 107

108

END of USA

Target: \\pssoft\pt8.45\bin\sqr\

mss\BINW\sqrw.exe?

HR/SA/SA-ZIN

### Program:

open a notepad file

Begin-program

print 'Welcome to SQR' (1,1)

End-program

Save the file with .SQR

Eg: "Demo.SQR"

## Execution:

### 2-Tier:

SQRW - Window

→ used for Developers.

→ SQRW.exe is the executable file for SQR.

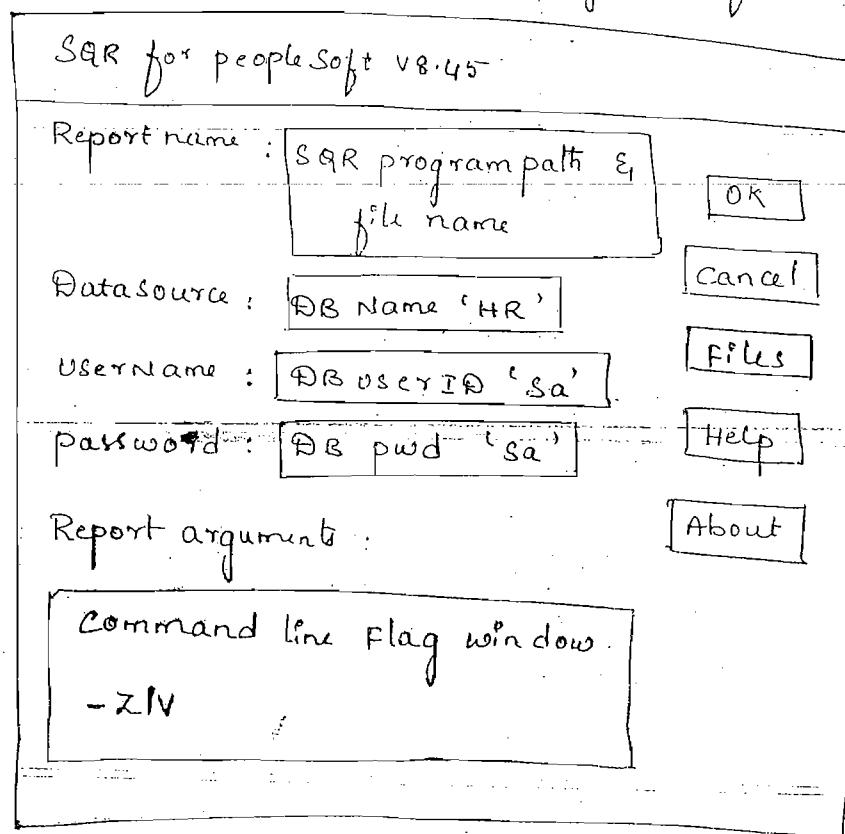
→ Path of SQRW.exe file

<drive>:\PT8.4X\bin\sqr\<Database Type>  
BINW\SQRW.exe

Eg:

E:\PT8.45\bin\sqr\mss\BINW

If we open SQRW.exe we get the foll. window.



click on files & go to the reg. path of file  
we saved.

→ Cannot open the printer file — This is the main error we get if already prev. output window is opened.

So close all o/p windows before executing the current program.

### Different File Types:

• SQR — Source code file

• LIS — List file / output file

• SPF — portable Format file

• SQC — Function library file

• INI — Initialisation file (config file for SQR)

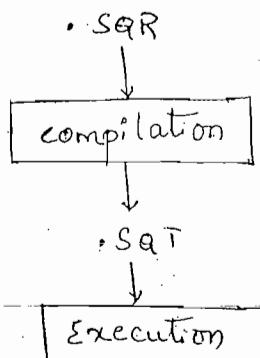
• SQT — Runtime file

↳ o/p after compilation is SQT

• MAX — Maximisation file



used to increase the SQR buffer space.



Temporary memory location to store some data for a point of time.

• HTML

• PDF

• CSV

• ERR — Error file (Error messages)

• LOG — Log file (Success & Error messages)

→ `ini - where` `^T` `^F` `J` `I` `I` `(7)`  
Soft

### Data Types :

- 1) Text / character
- 2) Date
- 3) Number
  - i) Integer
  - ii) Decimal

### Variables :

→ Temporary Memory location in SQR Buffer to store values.

#### i) Column Variables :

columns in a Table, Starts with & <variable name>

These are nothing but fields from a table in the

#### ii) SQR Variables :

PSDB ..

These are user defined variables created or starts with \$ or # defined by developers to store values.

#### • Text Variables :

\$ <variable name>

#### • Numeric Variables :

# <variable name>

#### • Date Variables :

\$ <variable name>

we cannot change these values in the variables  
↑ programmatically.

### iii) Reserved Variables:

System variables, starts with '-' (hyphen)

These are system variables, the values in these variables are maintained by the system.

### iv) Document Markers:

used in letters & document report types

starts with '@ <variable name>

These are the variables used in begin-document paragraphs to concatenate multiple values into a single variable.

### v) Substitution Variables:

These are the variables which are defined at the start of the prgm and that value remains unchanged in that variable thro' out the prgm.

starts with '{<variable name>}'

### Scope Variables:

Life time of variable

### vi) Global:

→ Default Scope

→ The life exists thro' out the SAR prgm.

ii) Local

Proof

→ No particular syntax → no need to declare variable

(8)

ii) Local :

→ The life exists only in that procedure where the variable is defined.

Syntax:

① Declare - variable

    Declare variable list

    End-declare

② Begin - procedure <procedure name> (<local  
variables>)  
    other commands

    end - procedure

## Program Constructs

i) if <condition> then  
    Statements  
    Else  
    Statements  
End-if

} used for conditional checking  
} used for single cond.

ii) Evaluate & variable  
    When = value  
    Statements Block  
    When = value  
    Statements Block  
    When - other  
    Statements  
End - Evaluate

} used for multiple conditional  
    checking  
} used to avoid writing multiple  
    'if' statements  
} we have to use block

iii) while & condition-  
Statements      { used for ! !  
                  used when the ending cond. is  
                  not known.  
End-while

→ 'FOR' loop is not supported by SQR, but it is implemented using 'Begin-Select'.

### Types of Reports:

#### i) Tabular Report:

Generating report from a single table

#### ii) Master-Detail Reports:

Generating report from multiple tables.

#### iii) Form Reports:

For each & every line separate page will be used

Eg: pay slip.

#### iv) Mailing Labels:

Used to partition slip vertically to effectively utilize the resources (paper)

Eg: Address printing

#### v) Letters & Documents:

in a report

Used to combine static text with column

Variables.

Eg: Bonus letters & Appraisal letters etc.

## v) Cross Tabular Reports

Used to perform row wise & column wise calculations.

### vii) Graphical Reports: Bar charts and

Used to represent data in a graphical format.

## Back end:

## Query Analyzer

↳ Select 'HR'

## To know Table names

App' designer

↳ file

↳ Open

↳ Select

Rec

Type select \* From ps-person Record.

Press F5 for execution.

Program: (Eg. for Tabular Report)

Open Notepad file.

Begin-Heading 3

print 'personal Details' (1,20) bold box

print 'EmployeeId' (3,1) bold underline

print 'Birthplace' (3,20) bold underline

print 'BirthCountry' (3,60) bold underline

End-Heading

Begin-Footing 2

print '\*\*confidential information\*\*' (2,10) bold

End-Footing

Begin-program

Do main ! Calling main procedure

End-program

Begin-procedure Main

Begin-select

emplID (+,1) → If we don't specify '+' the row values will not be incremented.

Birthplace ( , 20) all the opp values are

BirthCountry ( , 60) overloaded.

from ps-PERSON

end-select

end-procedure

Do :

→ Used to call a procedure

Syntax: Do <proc.name>

## Comments :

Not executable lines in a program.

Symbol used for commenting is !

Eg: !\*\* Start of Heading section.

Multiple lines commenting is not there. we have to use ! for each line.

## Master Detail Reports:

These are generated using foll. methods

- a) Using Joins
- b) Using procedures
  - i) Using global variables
  - ii) Using Local variables

### Using Joins:

Begin - program

Do Main

End - program

Begin - procedure main

Begin - Select

A. Emplid (+1,1)

A. Birthdate (,10)

B. Emplid

B. City (,45)

B. Country (,55)

From PS - person A, PS - Addresses B

where A.Emplid = B.Emplid

End - Select

End - procedure Main.

Eg: Using procedures (Global variables)

Begin - program

Do main

End - program

Begin - procedure Main

Begin - Select

A.Emplid (+1, 1)

A.Birthdate ( , 10)

move & A.Emplid To (\$Emplid)

Global variable

Do Addr-det.

From PS - person A

End - Select

End - procedure

Begin - procedure Addr-det

Begin - Select

B.Emplid

B.city (+1, 25)

B.country ( , 60)

From PS - Addresses B

move → used to move one value  
to another variable  
and for date type conversion

End -

End -

Move

→ C

→ D

→ E

→ F

Eq. U:

Begin

C

End

C

Begin

B(C)

A(C)

A(C)

C

C

Eric

EP

EP

EP

EP

Q

where IS-emplid = &emplid

(11)

End - Select

End - procedure

Move :

→ Used to move value from one variable to other.

→ used for data type conversion.

→ Syntax :

MOVE <Source Variable> TO <Destination Variable>

Eg: Using procedures (local variables)

Begin - program

Do main

End - program

Begin - procedure main

Begin - Select

A-emplid (71,1)

A-Birthdate ( , 10)

Do Empl-Det (&A-emplid)

From ps-Person A

End - Select

End - procedure

Begin - procedure Empl-Det (&Eid)

Begin - Select

If we use

Begin - Select Distinct then

it will give the data  
without duplicates.

need not work

Local variable



B. Emplid

B. city (+1, 25)

B. country (, 60)

From PS- Addresses B

where B. Emplid = \$Eid

And B. Address-Type = 'Home'

End - Select

End - procedure.

Form Reports program:

New-Page: for each & every row we will have a page

→ Used to open a new page for printing.

Begin-Heading 1 (0) BEGIN-PROCEDURE FORM-REPORT

print 'Form Report', (1, 20) Bold Box

End-Heading

begin-program

Do Form\_Report

End-program

Begin-Procedure Form\_Report

Begin-Select

Emplid

These are all retrieved from db & store

Name

it in SQR buffer.

monthly-Rt

Move & Emplid to \$Eid

Move & Name to \$ENAME

Move & monthly-Rt to \$Salary

Do printing

local variable

End

Begin

From PS - print Employee

(12)

End - Select

End - procedure

Begin - procedure printing

print 'Emplid : ' (1,1) Bold

print 'Name : ' (3,1) Bold

print 'Salary : ' (5,1) Bold

print \$Emplid(1,40)

print \$ENAME(3,40)

print \$Salary (5,40)

New - page

End - procedure

2/11/08

Cross Tabular Report Program

Cross tabular is used to perform rowwise and column wise calculations at a time.

Begin - Heading 1

print 'Emplid' (1,1) Bold underline

print 'Name' (1,5)

print 'Basic' (1,25)

print 'TA' (1,40)

print 'DA' (1,55)

print 'HRA' (1,40)

print 'Total' (1,85)

End - Heading

Begin - program

Do Main

End - program

Begin - procedure main

LET # Total-Basic = 0

LET # Total-HRA = 0

LET # Total-TA = 0

LET # Total-DA = 0

Begin - select

Emplid (+1, 1)

Name ( , 5)

Basic ( , 25)

TA ( , 40)

DA ( , 55)

HRA ( , 70)

move & BASIC to # Basic

move & TA to # TA

Move & DA to # DA

move & HRA to # HRA

Do cross-tabular

From PS-B14-Empl-Tbl

End - select

print 'Grand Totals' (+1, 1)

print # Total-Basic ( , 25) Bold

print # Total-TA ( , 40) Bold

print # Total-DA ( , 55) Bold

print # Total-HRA ( , 70) Bold

print # Grand-total ( , 85) Bold

# TBASIC = 0, 10, 30

# THRA = 0, 40, 70

# TTA = 0, 20, 50

# TDA = 0, 30, 70

# GTotal = 0, 100, 240

# Basic = 10 20

# TA = 20 30

# HRA = 30 40

# DA = 40 50

Emplid		TA	DA	HRA

End

Begin

Let

Let

Let

Let

Let

Let

End

LET

Sync

Mat

#

#

BS

End - procedure

Begin - procedure cross-tabular

Let # Total = # Basic + # TA + # HRA + # DA

Let # Total\_Basic = # Total\_Basic + # Basic

Let # Total\_TA = # Total\_TA + # TA

Let # Total\_DA = # Total\_DA + # DA

Let # Total\_HRA = # Total\_HRA + # HRA

Let # Grand-total = # Grand-total + # Total

Print # Total ( , 85)

End - procedure.

### LET:

Used to assign a const. to a variable.

Syntax: Used to insert data & also for retrieving  
to perform mathematical & string operating data.

Let # variable-name = value.

### Mailing Labels program:

# Define max-label-lines 10

# Define lines-between-labels 3

Begin - program

Do Labels

End - program

Begin - procedure Labels

Let # label-count = 0

Let # label-lines = 0

columns = 30 50

Begin - Select

EmpId (1,1,29)

Address2 (2,1,29)

Address3 (3,1,29)

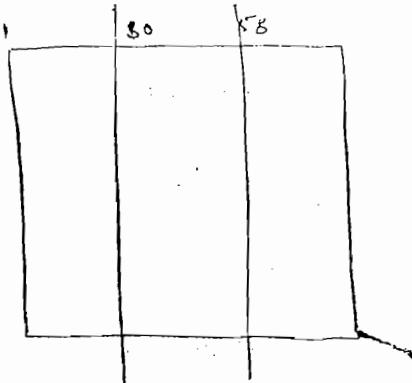
Address3

Address4

city

state

country



let \$addr-line = &Address3 || ',' || &Address4

let \$lastline = &city || ',' || &state || ',' || &country

print \$addr-line (4,1,29)

print \$lastline (5,1,29)

Next-column At-End = Newline

Add 1 to #Label-Count

if #current-column = 1

Add 1 to #Label-lines

if #Label-lines = {Max-Label-lines}

New-page

let #Label-lines = 0

else

Next-listing No-Advance skipLines = {lines-}

Between-labels }

end-if

end-if

From PS- Addresses

where Address-type = 'Home'

(14)

End-Select

Use-column # 0

New-page

print 'Total Labels printed : ' ( +1,1 )

print # Label-count ( ,30 )

End-procedure.

#Define :

Used to define substitution variables

Syntax:

# Define < sub-variable name > < value >  
{ < sub-variable Name > }

Columns :

Used to perform vertical partition of dlp page

Syntax:

Columns <c1> <c2> <c3> ...

|| :

Used to concatenate string values or variables

Syntax:

string1 || string2

LIE F. \$ <variable> = \$ <var1> || \$ <var2> || ...

Next-column : AT-END = NEW LINE

Used to move to next vertical partition  
after reaching the end of the column jump to the next line

Newline :

Used to move to next line.

## # Current - Column :

Stores the column no. where the cursor is currently located. Add `<n> GO # <variable>`

used to increment the variable value by `<n>`

then it is equal to `LET # <variable> <n> + 1`

NEXT-LISTING NO-ADVANC SKIPLINES = <n>

## Skiplines :

Used to leave empty lines This specification continues with next line of data by skipping the "n" number of lines on the output page.

## User - columns :

Used to exit from vertical partitions.

## Letters & Documents program :

.B — leave a blank line

## position () :

Used to assign value to document marker variable.

## Begin - program

Do BonusLetter

## End - program

Begin - procedure BonusLetter

## Begin - Select

Name

Address1

Address2

City

State

Country

Do Letter

From

End -

End -

Begin

Begin

Next

From ps - Employee

(15)

End - Select

End - procedure

Begin - procedure letter

Begin - Document (1,1)

{ & Name

& Address 1

& Address 2

@<sup>spare</sup> city - state - Country

.B

.B

Dear & Name

.B

This is to say you that.

.B

Regards

xyz

End - document

position(''@<sup>spare</sup> city - state - Country')

print & city()

print ','

print & state()

print ','()

print & country()

New - page

End - procedure

Viji  
SRNagar  
Ameerpet  
Hyd, AP, Ind

Dear viji

This is to

Regards,

xyz.

## Error Handling

→ Compilation stage

Begin-SQL on-error = skip/warn/stop

→ Execution stage

Begin-Select on-error = proc-name

Begin-SQL on-error = procedure-name

$-o$  : log file (.log file)      } starts with '-'  
 $-E$  : error file (.err file)      } command line  
 $-ZMF$  : error file (.dat file)      } flags

• SQL → Program execution sequence

↳ Compilation

Execution

- 1) checks for syntaxes      1) executes Heading section
- 2) checks database conn?      2) executes footing section
- 3) executes Setup section      3) executes program section
- 4) executes procedure section

→ After  $-o$ ,  $-E$  we have to specify the path

Eg:  $-o c:\Demo.$

Usage: while compilation & execution we have to type after  $-ZIV$  by

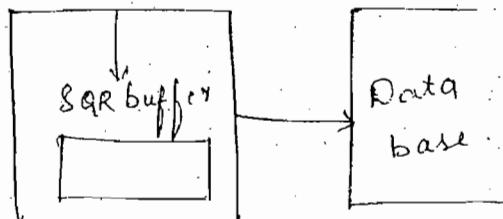
giving a space ex:  $-ZIV -o -E -ZMF$

## Load-Lookup (7)

(6)

- Used to increase performance.
- This is an array (variable used to store multiple values of same data type).
- We can only select two fields, but by string we can select more than 2 fields.
- Used in setup & procedure sections.
- Size of the load lookup will be increased automatically.
- Data gets populated during compilation stage.
- Doesn't support all datatypes, supports only character datatype.
- Data in load lookup gets populated using variable declaration command.
- The data that is populated will be stored in SQR buffer. So accessing SQR buffer is easier than accessing db. Like this it increases the performance coz ~~load-lookup~~ is a variable.

SQR



Concatenating means stringing

Breaking means unstringing

Advantages:

Used to create a load lookup array.

Syntax:

Load-lookup

Name = <load lookup name>

Rows = <no. of rows> [size of Load LOOKUP]

Table = <Table list>

Key = <key field from table>

Return Value = <other field from table> /concatenate field

Where = <condition>

Lookup:

to obtain return

Used to get the values from load lookup

Syntax:

Lookup <load lookup name> <if key value> \$  
  <variables>

Unstring:

Used to break a big string into pieces

Syntax:

Unstring <variable> <separator> <var1> <var2> ..

Program

Begin - setup

Load - lookup

Name = Empl-addr

Rows = 100

Table = PS - Addresses

Key = Emplid

Return - value = country ||| '#' ||| CITY

End - setup

Begin - program

Do main

End - program

Begin - procedure main

Begin - select

Emplid (+,1)

Birth date (+,5) work for return value

Lookup forward Empl-addr & Emplid \$value  
    <sub>(or #)</sub>

! unstring \$value #' into \$Eid \$Name

print \$value (+,50)

From ps - person

End - select

End - procedure

## Arrays: ①

- Used to increase performance.
- Arrays are variables used to store multiple values of same data type.
- We can select multiple fields.
- Used in setup & procedure sections.
- Size of the array cannot be increased automatically.
- Array supports all data types.
- Array get populated during execution stage.  
we have array commands used to insert data into array program.

## Create - Array :

Used to declare an array variable

### Syntax:

Create - Array

Name = <Array name>

Size = <array size>

Field = <field names : <data type>

Field =

: , : , : ,

: , : , : ,

: , : , : ,

### Set:

Used to insert values into any array as well as  
retrieve value from array.

Syntax

(18)

a) For inserting

Let <arrayname>.<fieldname>(<index>) =  
variable / value \$ / #,

b) Retrieving

Let \$ / # <variable> = <arrayname>.<fieldname>  
(<index>)

Put : Let, put, get are SQL "array commands"

Used to insert data into an array.

Get :

Used to retrieve data from an array.

Begin - setup

Create - Array (used to declare an array variable)

Name = Employee

Size = 1

Field = Emplid : char

Field = Name : char

Field = Basic : Number

Field = TA : Number

Field = DA : Number

Field = HRA : Number

Field = Total : Number

End - Setup

Begin - Heading 1

Print 'Eid' (1,1) Bold underline

print 'Name' (1, 5)

print 'Basic' (1, 25)

print 'TA' (1, 70)

print 'DA' (1, 90)

print 'HRA' (1, 110)

print 'Total' (1, 130)

end-Heading

Begin-program

Do inserting

Do printing

End-program

Begin procedure inserting

Begin-select

Emplid

Name

Basic

TA

DA

HRA

Let #TOT = & Basic + & TA + & DA + & HRA

Let #I = #I + 1

Let Employee. Emplid (#I) = & Emplid

Let Employee. Name (#I) = & Name

Let Employee. Basic (#I) = & Basic

Let Employee. TA (#I) = & TA

let Employee ~ (44 - ) - ~ ~

let Employee · HRA (#I) = & HRA

let Employee · Total (#I) = #TOT

From ps - BH - Empl Tbl

End Select

End procedure

Begin-procedure printing

let # count = # I

let # I = 1

while # I <= # Count

let \$emplid = Employee · Emplid (#I)

let \$Name = Employee · Name (#?)

let \$Basic = Employee · Basic (#I)

let \$TA = Employee · TA (#I)

let \$DA = Employee · DA (#I)

let \$HRA = Employee · HRA (#I)

Let # Total = Employee · Total (#I)

print \$emplid (1,1)

print \$Name (,5)

print \$Basic (,25,10)

print \$TA (,60,10)

print \$DA (,75,10)

print \$HRA (,190,10)

print #Total (,105,10)

$\# I = \# I + 1$

end-while

end-procedure

Runtime variable

→ To provide i/p parameters for 2-tier we use

ASK & input

→ To provide " " " in " "  
Run control tables.

Runtime Variables (Q)

Ask:

→ Used in Setup section (main uses in 'Begin-Setup')

→ This prompts the value during compilation stage

→ No datatype, No length, No formatting

Input:

→ Used in all sections except setup

→ This prompts the value during execution stage

→ we can specify datatype, length and format

Syntax:

i) Ask:

a) ASK <variable name> <information text>

b) '{ask variable}'

P.T.O.

Bel:

A

E

Test

→ -P

→ L

→ -C

→ T

→ O

→ P

→ C

→ C

→ C

→ C

→ C

→ C

→ C

→ C

→ C

→ C

→ C

Input:

- a) input \$/# <variable name> <information text> (2)
- b) \$/#>input variable > where empid = \$empid

Program: (Joins prgm)

Begin-Setup

Ask entry 'Enter country'

End-Setup.

Begin-Select

From ps-Addresses

where country

'entry'

Generates 3 pages.

Testing & Debugging: (I)

→ -Tnn : pages } used for testing purpose. T3

→ Loops = nn

Begin-Select loops = 100.

→ -Debug

→ Display: one variable

→ Show: More than one variable

→ ## if ref Debugx → we can use any character

Display <variable>

# End -if

Normal

debugging

conditional

debugging

used for debugging  
purpose.

→ Hard errors means syntax errors

soft errors means programming errors.

→ Debugging is used to find programming  
errors.

→ Display is used for single col. only whereas  
Show is used for multiple col. at a time.

In Tabular Report prgm write the foll in main prgld

Begin-Select

Display &emplid

Display & Birthcountry

Like wise do for Show also

Show &emplid & Birthcountry

→ If we write this after End-Select it displays  
only last Emp. details.

→ After Birthcountry (,60) write

#ifdef Debugx

Display &emplid

#endif.

#ifndef debugy

Show &emplid & Birthcountry

#endif.

-ZIV -Debugx → Use this command line flag

-ZIV -Debugy

-ZIV -Debugxy → both x, y debugs will execute

#ifdef debugx

Display &B38-emplid

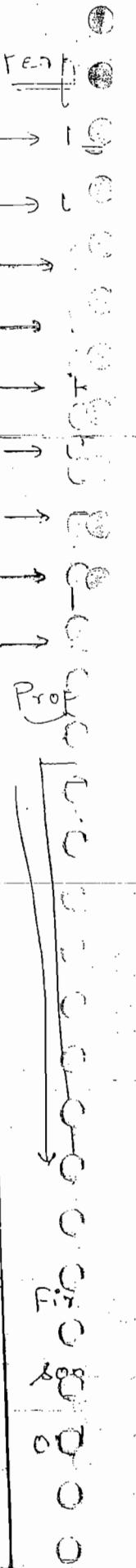
#endif

#ifdef debugy<sup>→ no spcl.</sup>

Show &B38-Name &B38-Basic

#endif IF

conditional  
debugging



## Performance Tuning Impl

(2)

→ Load Lookup, Lookup.

→ Using Arrays.

→ cross tabular reports

→ Multiple reports

→ Running on server

→ Using SQL files

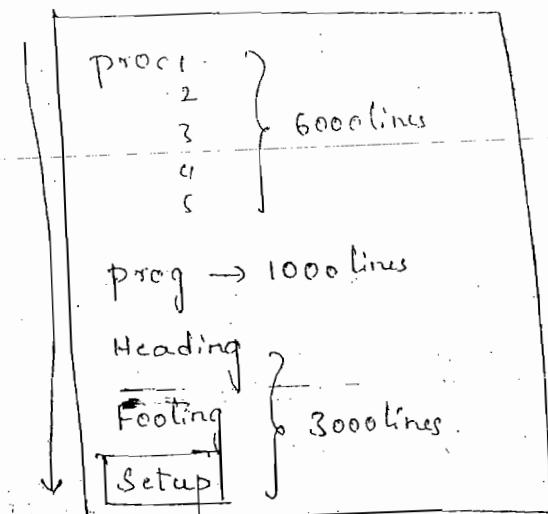
→ -Bnn -B50 → No. of trips will be reduced while retrieving data from

→ proper programming logic. PSQB to SQR buffer.

→ SQL tuning. Bulk operation.

Proper programming logic.

SQR 10,000 lines



Setup  
Heading  
Footing  
program  
procedure

proper order

Here we have written setup section at the last.  
First system searches for setup " so it takes  
some time for it. If we write in a proper  
order we can reduce the time for searching

Emp-Tbl

Pid	Eid	Name	Sal

Dependent table

Pid	Eid	Dept-id	Name	age

Begin-Select

A.Eid (+1, 1)

A.Did (., 5)

A.Name (., 15)

B.Dept-id (., 40)

B.Name (., 60)

B.age (., 70)

From ps-Emp-Tbl A, ps-Depen-Tbl B

where A.Did = B.Did

And A.Eid = B.Eid.

1) Match the key fields

2) place child table on left hand side } By doing like  
this performance will increase.

→ The procedures which we have written in a SQR prgm cannot be used in another SQR prgm, but by using procedures in SQC files

we can use those procedures in SQR prgm

1) Select

F

1) Env?

Eg

2) Set.

Env

3) Proc

Tr

C

H

C

C

E

O

O

O

O

## SQCs:

22

Function Library file

### 1) Environment SQCs:

Used to make SQR OS & database independent.

Eg: setenv.sqc

### 2) Setup Section SQCs:

① Consists of setup section code.

Eg: setup31.sqc, setup01.sqc - payroll

setup32.sqc, setup02.sqc - landscape

### 3) Program Section SQCs:

Consists of program section code.

This looks like xxprogxx.sqc

Eg: HRProg31.sqc

### 4) Header SQCs:

Consists of Heading section coding

looks like xxHdgxx.sqr

Eg: HRHdg31.sqr

### 5) Footer SQCs:

Consists of Footing section coding

xxFigxx.sqr

Eg: Reset.sqr

### 6) API Aware SQCs:

Used to execute SQR programs on Peoplesoft

Batch server

Eg: stdapi.sqc

7) Ask SQCs:

consists of Ask/Input Commands

8) Run Control SQCs:

Consists of run control parameters coding in n-tier.

9) Functional SQCs: (datetime.sqc etc)

Used to support diff formats to data

10) module specific SQCs:

consists of module specific coding

Command Line Flags:

→ -Bnn - Bulk operation (.SPF + -RS + -RT)

→ -ZIV - .SPF, .LIS, .SQT → These are all created in the same place of .SGR file

→ -E - error file .err

→ -O - log file .log

→ -C - used to see a cancel button during prgm execution. Eg: -ZIV -C

→ -P - used to specify output file path

→ -KS → SGT used to compile (.SPL)

→ -RT - SGT used to execute (.LIS)

→ -Tnn - No. of o/p pages in testing

On-Break Eq:

① Begin = program ② without levels.

③ Begin - select

Emplid (+,1)

Ename (,10) on-Break

Country (,20) on-Break

From PS-Record\_Empl

End - Select

End - program

O/P:

101 ANU, ♂ IND

123

34

567 USA

104 ARUN, ♂ IND

105 KIRAN, ♂

102 SUNI, ♀

234

2343 USA

24 IND

323

4344 PAK

435 USA

AWE

IND

103 VIJAYA, ♀

676

cy  
Bec

D

Ex

Beg

#

E

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

b) With levels

Begin - program

Begin - Select

Emplid (+1,1)

Ename (,,10) on-Break Level = 1

country (,,20) on-Break Level = 2

From PS-Record-Empf

End - Select

End - program

O/P:

101 ANU, ♀ IND

123

34

567

USA

104 ARUN, ♀ IND

105 KIRON, ♀ IND

102 SUNI, ♀ IND

234

USA

2343

IND

24

323

4344

PAK

USA

435

IND

AWE

IND

103

VIJAYA, ♀

## ~~29~~ Debugging

24

Begin-program

Do Main

End-program

Begin-procedure Main

# IFDEF DEBUGY

Begin-select

Emplid (+1,5)

Op will not appear

Ename (,30)

for -ZIV -DEBUGX

# IFDEF DEBUGX

also

DISPLAY & Emplid

# END-IF

SHOW & Emplid & Ename

FROM PS\_SQL\_Empl\_Tbl

End-select

# end-if

End-procedure

If we not specify -ZIV -Debugy op will not appear.

-ZIV

If we give -ZIV -Debugy, we get the op on  
The SAR viewer as well as shortcut to Sproc  
window

If we specify -ziv -DEBUGLY, both debugs will execute simultaneously.

### Display:

Begin-program

Do main

End-program

Begin-procedure Main

Begin-select

Emplid (+1,5)

country (,30)

Display & Emplid

Display & country

From ps-addresses

End-select

End-procedure

Displays both Emplid, country simultaneously on sgrw.exe

### Show:

Begin-program

Do main

End-program

Begin-procedure Main

Begin-select

Empi  
Ena  
St

Eni  
Eng

16/11/08  
File

) O  
) G  
) C  
) D  
) E  
) F

i) cl  
3) Re  
4) w

syn  
eg  
eg  
eg  
eg

bugs

Emplid (+1, 5)

Ename (, 30)

(25)

Show & Emplid & Ename

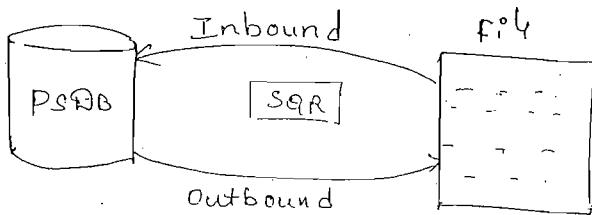
From ps-SQL-Empl-Tbl

End-select

End-procedure.

16/11/08

## File Handling Commands: (I)



i) Open: used to open a file to perform operation  
↓  
writing data, read data, to append data.

- ii) close: used to close a file.
- iii) Read: used to read data from file and copy into a variable. For Inbound we used read.
- iv) write: used to write data into a file. For Outbound we used write

Syntax:

- i) Open <file path> As <alias number>  
FOR-Reading / FOR-writing / FOR-Append Record: <n>  
Status = <numeric variable>

② close <filepath/alias no>

③ write <Alias no/ file path> from <variable1[:length]>  
<variable2[:length]>

④ Read <Alias no> into \$<variable>[:length]  
file path

### Scheduling SQR / Running on Server (n-tier execution)

I) R

i.) Make SQR API Aware

    a) Include foll. SQR's

        i) stdapi.sqc : make SQR to run on PS systems

        ii) setenv.sqc : make SQR portable to run on  
                     any DB & OS.

    b) Call the foll. procedures from stdapi.sqc

        i) stdapi-init : called during start of

                  execution stage

        ii) stdapi-term : called just before the end of  
                  execution stage

2) Move .SQR file (source code file) to

    <Drive>:|PT 8.4X|SQR

3) Create required run control definitions

    a) Run control Table : used to run any process/  
report in n-tier, used to pass i/p parameters to  
process/report

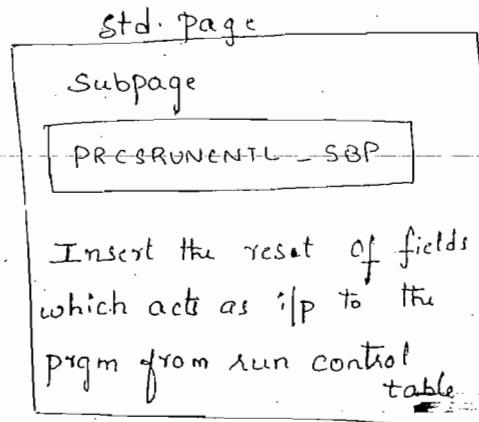
conditions to be defined for querying

- i) OPRID & RUN-NTRN CNTL-ID must be high  
level key fields.

(2b)

Created only if i/p parameters are required  
to run process/Report

- b) Run control page:



root of run ctrl page has subpage.

- c) Run Control Component:

Insert page : Run control page

Search record : PSPRCRSRUNENTL

&

Register the component.

H) Create Process definition: Used To Link the program

with run control component

(1) R

Nav!

People Tools → Process Scheduler → Processes

Eg:

For Tabular report program

① # include 'std stdenv.sqc' → at starting

# include 'std api.sqc' → at last

② Begin-program

do stdapi init

Do main

do stdapi term

End-program

③ E: | PT8.45 | <sup>sqc</sup> / Paste Tabular program here (bassq)

④ we don't have any I/P parameters

so no need of run ctrltbl

Select a page

Insert → Subpage

↳ PRCSRUNCNTL-SBP

OK

Program

Save workspace

(2+)

Save project B25-SQR-PROJECT

Select a component

Insert B25-SQR-RUN-PG.

Search Record: PSPRCRSRUNCNTL

Save B25-SQR-RUN-CMPT

Register the component

⑤ IE → Go to our comp.

↳ Run → Here we can't able to  
see the process name &  
process type.

People Tools → process scheduler

↳ processes

Add a New

Process Type: SQR Report

Process Name: b25sqry

OK

API Aware

Go to process def. page

Comp: B25-SQR-RUN-CMPT

Process Groups: All pages

All panels

Save ↵

Now Go To our comp

↳ Now process type & process name will  
be appeared on the page.

Page

Seq

Re

He

Co

C

C

C

C

C

C

C

#.16

Begin

C

C

C

C

End

Beg

Beg

Ep

O

O

O

Go To process Monitor.



See the run status

Refresh



Eg: Using I/P parameters.

Tabular prgm

1. Create a run control Table 'B25-SQR-RUN-TBL'

OPRID K,

Row init event:

B25-SQR-RUN-TBL-OPRID-VALUE =

PSPRCSTRUNCNTL-OPRID-value;

RUN-CNTL-ID K,

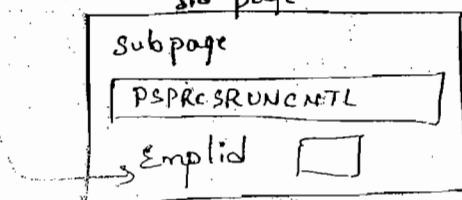
Row init event:

B25-SQR-RUN-TBL-RUN-CNTL-ID:value =

PSPRCSTRUNCNTL-RUN-CNTL-ID-value;

Emplid:

2. Create Run control page 'B25-SQR-PG' std.page



3. Create run control component

(28)

Page : B25-SQR-PS

Search record PSPRCSTRUNCNTL

§

Register Component

PSPRCSTRUNCNTL

OPRJD	RUN-CNTL-ID.
PS	
PXS25	Test

B25-SQR-RUN-TBL

OPRJD	RUN-CNTL-ID	Emplid
PS	1	101
VPI	Test	102
PXS25	Test	110

§ PRCS-OPR ID :

§ PRCS-RUN-CNTL-ID =

§ Emplid = 110 .

#.include 'setenv.sqc'

Begin - program

Do stdapi\_init

Do param

Do Tabular

Do stdapi\_term

End - program

Begin - procedure Tabular

Begin - select

Emplid (+1,1)

Name ( ,5)

Basic ( ,40)

From ps-Empl-Prm

where Emplid = \$ Eid

End-procedure

Begin-procedure param

Begin-select

OPRID

RUN-CNTL-ID

Emplid

move & Emplid to \$eid

From ps-BAS-SQR-RUN-TBL

where OPRID = \$ PRCS-OPRID

AND RUN-CNTL-ID = \$ PRCS-RUN-CNTL-ID

End-select

End-procedure

# include 'stdapi.sqc'

(1)

→ \$QC's related to setup section are included

on the top of prgm & \$QC's not related

to setup section are included at the bottom

of SQR prgm.

! MULL

→ E7

tim

-n

→ Usp

o/v



thus

whil

from

data

(1)

Comn

) DQ

→ U

→ US

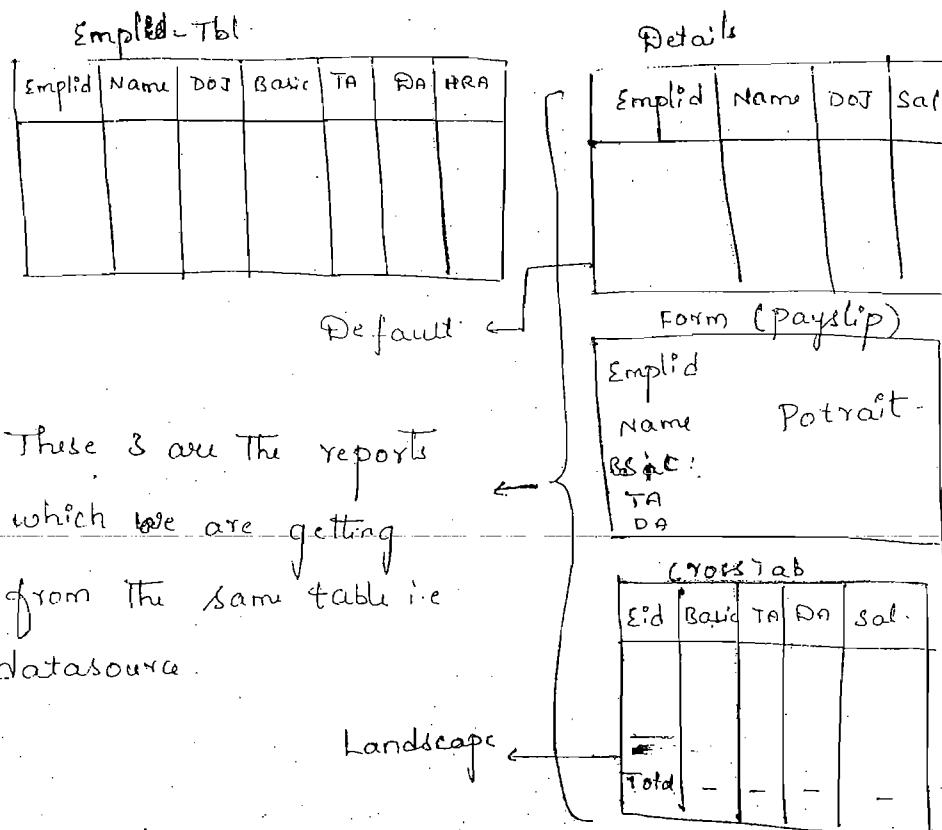
dyn

(1)

## Multipurpose reports

(2)

- Executing the same program in less amount of time by doing some changes is called performance tuning.
- Used from the same datasource multiple off-s are created.



## Commands:

### 1) DECLARE-LAYOUT:

→ Used in setup section.

→ Used to specify the list of layouts used.

## Syntax:

DECLARE-LAYOUT < Layout Name >

Other Commands

End-declare :

Eq: Declare-Layout Emp-Det

End-declare

Declare-Layout Pay

# include 'setup31.sqc'

↓  
sqc used for portrait format

End-declare

Declare-Layout crts

# include 'setup32.sqc'

End-declare .

## 2) Declare-Report:

→ Used in setup section

→ Used to assign layouts to reports

→ Depending upon no. of o/p's we are generating

that many declare com report commands we

should have in our SQR program.

Syntax:

Declare-report <Report Name>

Layout = <Layout Name>

End-declare .

→ Layout name & layout  
diff.

(30)

Ex: Declare-report Emp-Act

Layout = 'Emp-Act'

End-declare

Declare-report paylip

Layout = 'pay'

End-declare

Declare-report crossTab

Layout = 'cross'

End-declare

### 3) For-report:

[If we want headings & footings for reports]

Diff. headings & footings for diff. reports]

→ These are used to assign diff. headings &  
diff. footing sections to diff. reports

② → In case of multiple reports we can have  
more than one heading & footing sections.

Syntax:

Begin-Heading/Footing <n> for-Report = <report  
name>  
print commands

End-Heading/Footing

Eg:

Begin-Heading 2 For-report= Emp-Det

print 'Employee Detail Report' (1,1)

Bold underline

print 'Emplid' (2,1) Bold

print 'Name' (2,5) Bold

Emp Detail		
Employee Details		
Emplid	Name	DOJ

conf.info...xx

End-Heading

Begin-Heading 1 For-report= payslip

print 'payslip' (1,1) Bold

End-Heading

Begin-Heading 1 For-report= crossTab

Print '{Emplid}' (1,1) Bold

Payslip			
no sign. reg.			

End-Heading

Emplid	Name	TA	DA

Begin-footing 1 For-report= Emp-Det

print '\*\* conf.info.\*\*' (1,1) Bold center

End-Footing

Begin-Footing 1 For-report= payslip

print '\*\* no sign. reg. \*\*' (1,1)

End-Footing

#### 4) Use-report

(3)

→ This is used to assign the commands used in prgm or procedure sections to diff reports.

Eq:

Use-Report = payslip

Print & Basic (5,40)

Print & TA (7,40).

Use-Report = crosstab

Print & Basic (1,10)

Print & TA (1,20).

#### File Handling:

Eq:

#### DB2 file (outbound prgm):

Begin-program

Let \$writeln = "C:\temp\Batch25-scr.txt"

Open \$writeln As 30 for-writing Record:100

status = #filestat

if #filestat != 0

Print "Error in open file" (2,1) center bold

Else

Do write-Data

close 30

Print "Writing data is complete" (1,1)

End-if

End-program.

Begin-procedure write-data

Begin-select

Emplid

Name

Basic

TA

write 30 from &Emplid:5 &name:15

&Basic:10 &TA:10

From PS-B1H-Empl-Tbl

End-select

End-procedure.

Eg: Inbound program: (file2DB)

Begin-Setup

Begin-SQL

create table batch25-tbl (DeptId varchar(2),

Emplid varchar(3), Name varchar(20));

End-SQL

End-Setup

Begin-program

let \$writefile = 'c:\input.txt'

Do file handling

close 30

print 'insertion complete' (1,1) Bold (32)

End-program

Begin-procedure file handling

open \$writefile as 30 for - reading record=100  
status = #filistat

if #filistat != 0

print 'Error in open file' (2,1) center bold

else

while 30

read 30 into \$line :100

unstring \$line by ',' into \$DeptId  
\$EmplId \$name

No insertion

if #end - file

break

End-if

End-while

End-if

End-procedure

Begin-procedure insertion

Begin-SQL

Insert into batch25-Tbl (DeptId, EmplId,  
name) values (\$DeptId, \$EmplId, \$name);

End -SQL

End -procedure.

### N-Tier Execution program :

#include 'setenv.Sgc'

Begin -program

Do stdapi -Init

Do Param

Do Main

Do stdapi -Term

End -program.

Begin -procedure Main

Begin -Select

A. Stuid (+,1)

A. Sname ( , 30)

From PS -STURRECORD A

where A. stuid = \$ Empid

End -Select

End -procedure

Begin -procedure param

Begin -Select

B. OPRID

B. RUNCNTL-ID

B. STUID

Move A. STUID To \$ Empid

From PS -LL - PRCSRUNCNTL B

### NTIER PROJECT

#### Page controls:

STOPPROJECT

Where B-OPRID = \$ PRCS-OPRID

(23)

AND B-RUN-CNTL-ID = \$ PRCS-RUN-CNTL-ID

End-select

End-procedure

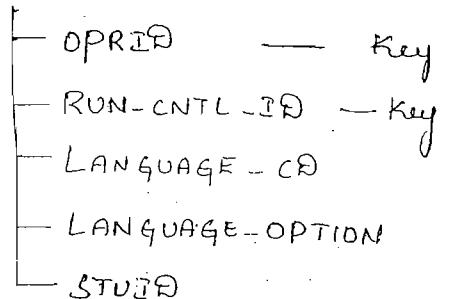
#include 'Stdapi.h'

#include 'StdDateTime.h'

#include 'Number.h'

App' Designer.

Record — LL-PRCSRUNCNTL



Page : PRCSRUNCNTL-SBP

STUDENT ID

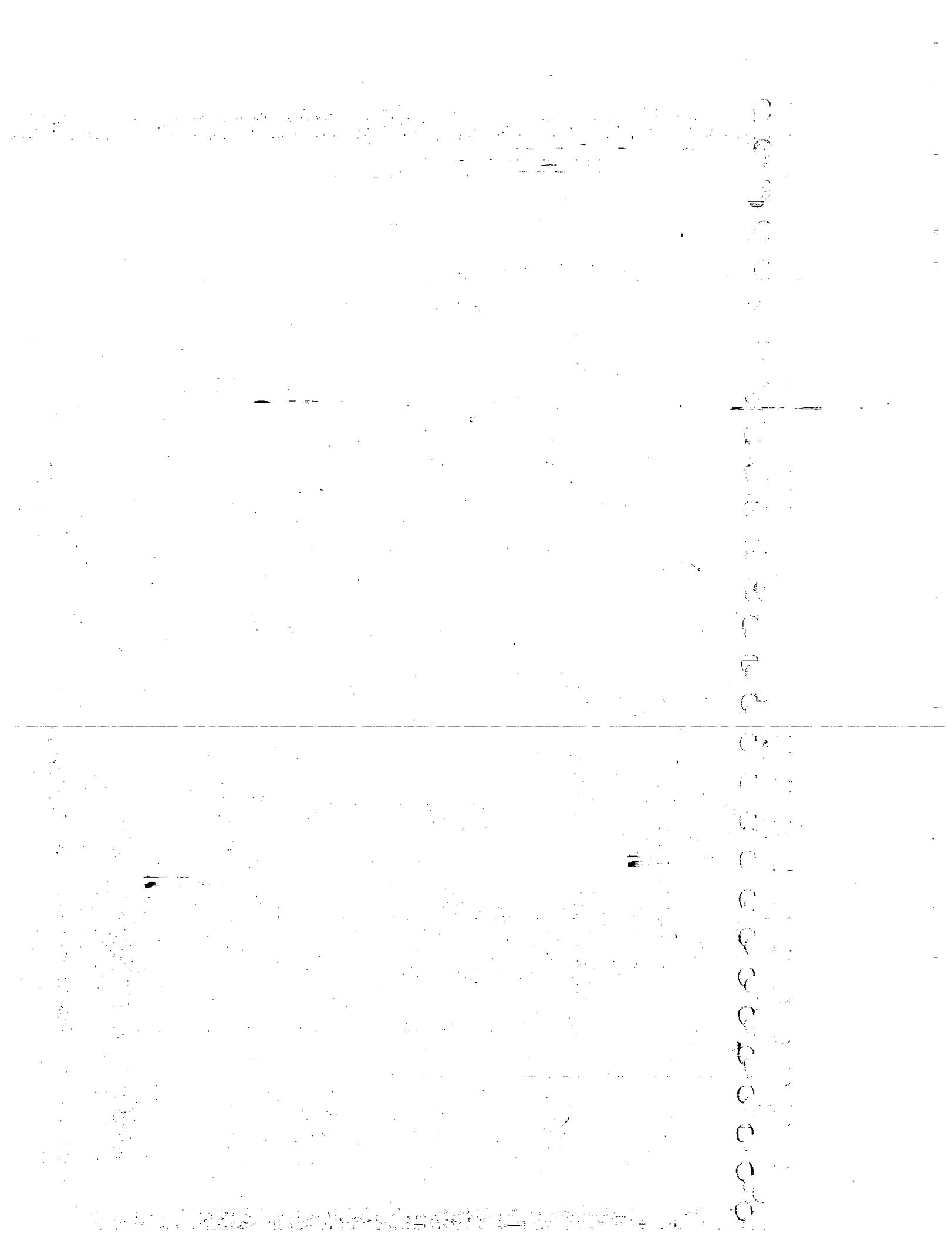
} LL-SUBPG

Component : — LL-comp

Search Record : PRCSRUNCNTL

Register the component.

The OPRID & RUN-CNTL-ID are values are captured in the table LL-PRCSRUNCNTL



28/11/08

## Crystal Reports

PS using many reporting tools.

- \* (1) SQR (Structured Query Reports)  
↳ also called as Brio Reports
- } 3rd party Tools.

(2) Crystal

\* (3) PSnvision

(4) PS Query

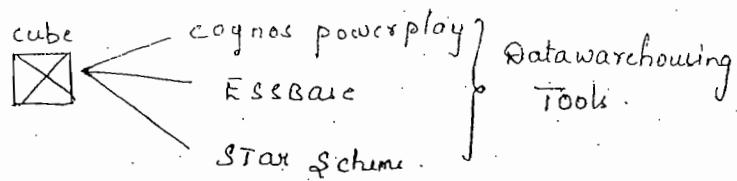
(5) XML publisher (Newly from 8.48)  
PT

(6) OLAP Tools.

↳ Tool name is Cube Manager.  
↓

Using this we can able to generate cubes.

Cubes contain data, From db we are populating  
data into cubes.



→ PSnvision is mainly used for financial GL module.

→ PS Query is source for PSnvision & Crystal reports

→ without PS Query we cannot generate " "

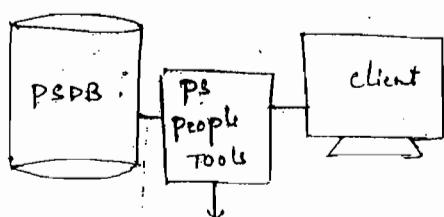
→ with the support of Datawarehouse tools we  
can generate multi dimensional tools using  
Cube Manager.

→ Advantages of cube  
1) Security

→ ps provides security at Frontend level.

### Crystal Report:

→ Third party Tool.

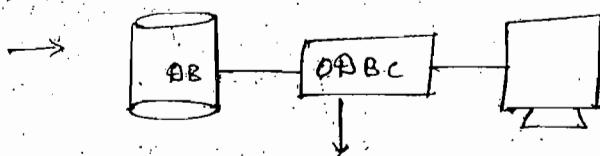


ODBC provided by peoplesoft & this we can use with ps while working with crystal reports.

This ps people tools connected to Frontend.

→ In SQR there is no security. While generating reports we can provide security.

→ This ps people Tools ODBC can access queries

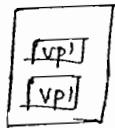


This ODBC can access tables & views.

This ODBC connected to backend we are providing userid & password of db



(3) But where as for ps peopleTools ODBC we are providing user ID & password of PSDB.



→ ps peopleTools is going to connect from front end so that security can be enabled.

### Navigation:

programs → crystal reports 8.5 for peoplesoft

↓  
crystal reports for people soft

welcome to crystal Reports.

create a New crystal report Doc.

① using the Report Expert

② As a Blank Report.

If we select this it displays some type -

= Select the req. one.



① click on Database

↳ ODBC

↳ FIN.

↳ Login id : sa  
password : sa

Select the req. Table

FIN.dbo.ps - ACCOMP-TBL

click on **Fields**

↳ Select the req. fields for report generation

**Select** Tab

↳ used for where clause

**Finish**

→ 3 Security Tables

(2) ODBC

↳ peopleSoft peopleTools

userid : vpt

password : vpt

Select the req. Query

→ These reports are generated from outside peopleSoft system.

Steps required to Design & Run Crystal Reports From peopleSoft APP

① Define Query. Define the crystal report by using peopleSoft PeopleTools ODBC.

(B)

② Define Runcontrol Table (or) use existing runcontrol Table, Define Runcontrol page, component and register the component.

③ Schedule the crystal reports by process scheduler. Authorize the user to run the report.

④ Test Run.

① Open App' Designer.

↳ Query.

choose the req. Table:

↳ JRNL-IN.

(Double click on it).

Select the req. fields for report.

provide the where condition in Criteria tab.

Run the report.

BU : US001
Ledger : Local
Currency : US
Dept : 20000

Save.

Owner : public Type : Query

Query Name : B25-crystal

Description : Journal Report.

Generate the crystal report as discussed before  
with Query we selected created.

- Create Run control Table, page & component
- Register the comp.
- Add comp to process Scheduler.

Override options

Parameters list

- ORIENTP: RUN-B44-JRNL.BUSINESSUNIT:

[Write the recordname • Field Name]

29/11/08

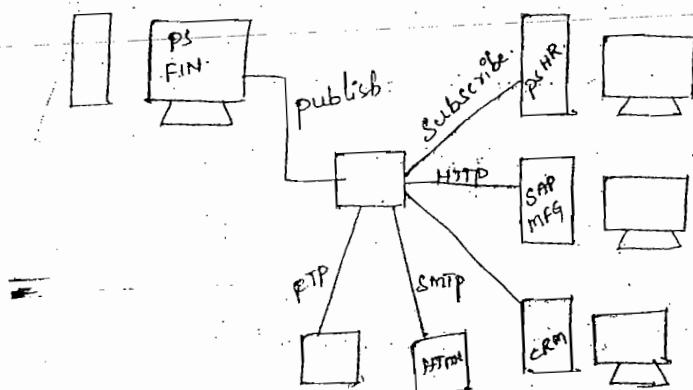
## Integration Tools

(3)

→ ps Integration Tools are divided as follows.

	INBOUND	OUTBOUND	
Synchronous (Real Time)	component if, integration broker	Integration Broker,	Business interlinks (Before version)
Asynchronous (Near Real time)	IB	IB	
File Based	File Layout/AE /CII/people code	File Layout/AE /people code	

### Integration Broker:



→ One sys. publish data & all other sys. subscriber data

Eg: PS FIN, publishes Account code all the 3 sys.  
Subscribes that account code

→ Sync. — one sys publishes & only one sys can subscribe.  
Eq: Telephone call (One to one)

Async — one sys. publishes & many sys. can subscribe.

Eq: Email (one to many)

→ Sync - Response is must, means if a sys. publishes then one sys. can subscribe & it has to send response back.

Async - Response is not required.

→ Comm<sup>n</sup> is always happen in XML.

→ IB have a component which takes info. to people soft & change the protocol according to the sys. that is subscribing.

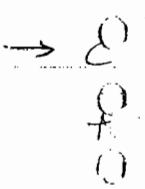
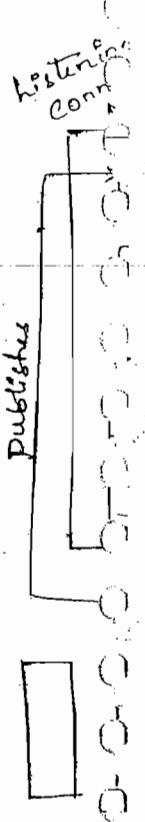
[i.e SMTP, HTTP...]

→ IB is divided into 2 components:

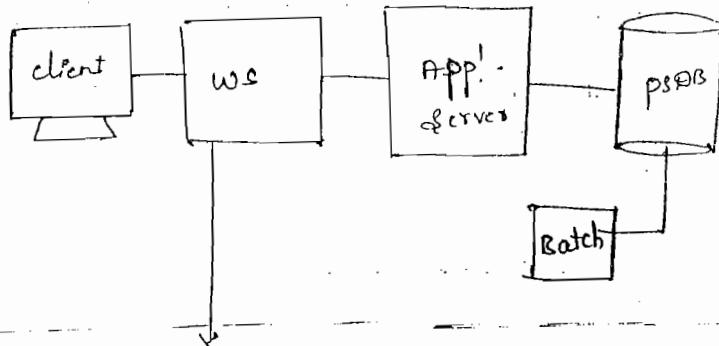
① Integration Engine

② " Gateway — is a servlet & resides in webserver.

→ Integration Engine resides on App'server

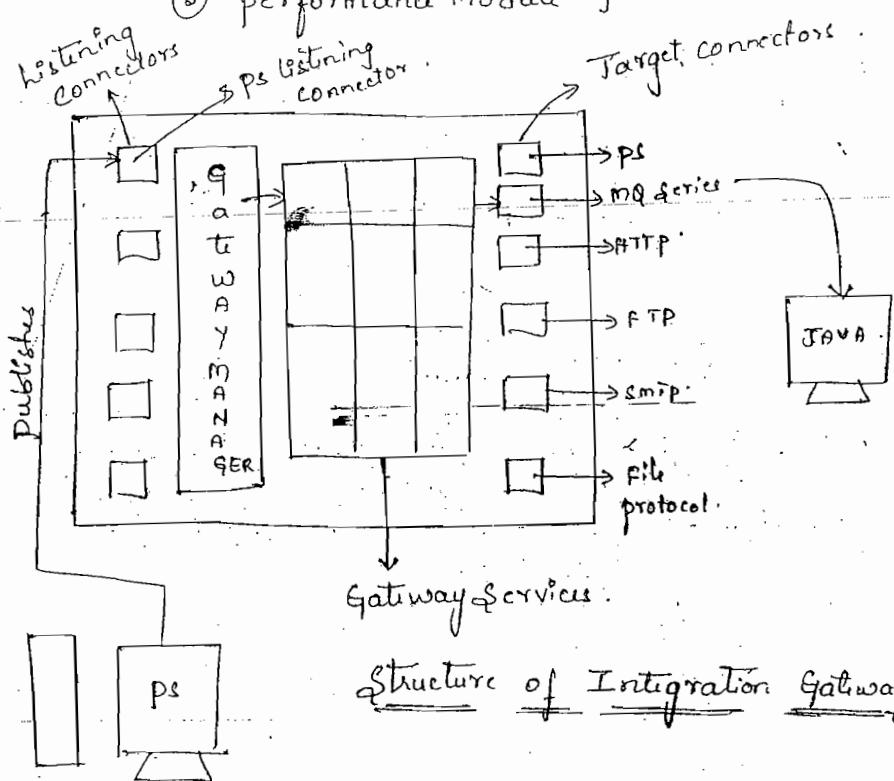


(3)



- ① Portal servlet
- ② Report Repository
- ③ Integration Bridge
- ④ Search servlet
- ⑤ performance module

{ servlet resides in  
webserver.



→ Connector Development Kit → is a software using  
this we can develop our own connectors.

J2EE

→ we can register the connector in the Gateway.

→ Before 8.44 Integration Broker is called as App message.

8.0 - 8.19 → App' Message

8.40 - 8.47 → IB

8.48 - XX → IB.

### Integration Engine :

- ① Message
  - ② Message channel
  - ③ people code.
    - ↳ ④ Sync. inbound.
    - ⑤ Sync. outbound.
    - ⑥ Async. inbound.
    - ⑦ Async. outbound.
  - ⑧ App' engine → program type, must be transform.
  - ⑨ Code sets → The config. of code sets is taken care by the developer.
  - ⑩ NODE Definition.
  - ⑪ Gateway " "
  - ⑫ PUBSUB Servers
  - ⑬ Integration gateway properties
- Take care by developer
- Take care by Admin people

→ Unless pubsub servers started we cannot integrate one sys. to another sys.

→ If we want to define msg first we have to define msg. channel, b'coz while defining msg we have to assign msg. channel otherwise we can't save msg.

File → New

↳ message channel.

Save → B4 - msgchnl.

Go to properties of msg. chnl.

Use Tab.

message channel status

RUN → must be Run → Active.

PAUSE

↳ Inactive.

Archive messages ?

unordered

↳ If we select this each msg is independent

File → New

↳ message

Insert → child Record

↳ GL\_Account-TBL (Select)

Select GL-Account-TBL

↳ Select child rec.

Insert → child record.

↳ Any Table ]

Save → B24-Account.

Go to prop. →  Use Tab.

↳ Message channel : B24-mqchnl  
version

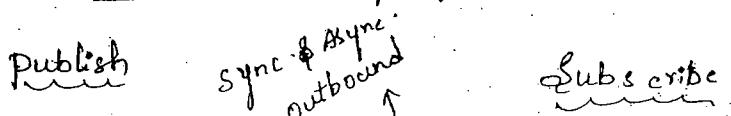
Message Subscriptions — we have to write the people code.

Insert Message Subscription.

↳  Use Tab.

↳ Account.

Events used to write people code in IB:



- ① Record field → save post  
Change / Field change
- ② component → "
- ③ AE → On Execute

- ④ Message
  - ① @Subscription
  - ② on Routigend
  - ③ on Route Receive

(4)

① On send

② On ACK, receive

③ On Request

→ Subscription - Async.  
inbound

- on Request - syn inbound.

On Route & end - publishing

side filtering.

On Route, Receive - subscribing

side filtering

Service operation contains events

In the New versions we write these events in  
App class.

App package.

↳ contains classes

↳ contains code.

→ classes used in IB.

① Message class

② XML doc / Node

③ SOAP class → mainly using web msgs.

comm' is point to point

we are offering web service.

Any to local, local to local.

→ msg class is divided into ~~a~~ types

(a) Rowset msg

(b) Non-Rowset Msg.

Rowset msg format is valid only to ps to ps.

Non " " " " " " to ps to any sys.

↳ msg class is not valid; we use xml doc.

To Non people soft — XML doc.

To people soft — msg class.

→ Declare variable.

Instantiate the object

calling

AE: Requirements to use AE in IR

Transform

} Go for xslt [xml] action.

Translation

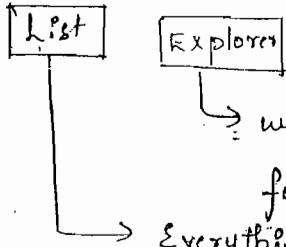
Filtrering — " " people code action.

Translating values of fields

## Report Manager :

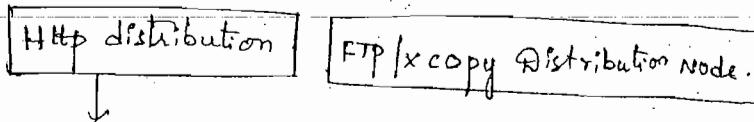
### • Reporting Tools

↳ Report.mgr ..



while working with process monitor & Report mgr  
when we click on link Report Repository <sup>service</sup> opens the  
particular report & generates it.  
process scheduler.

↳ Report Nodes

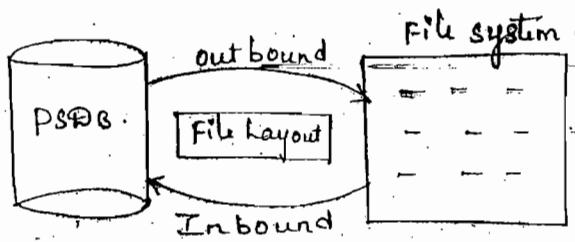


Best way

## File Layout

(2)

- This is a proprietary Tool.
- " " an integration tool.
- File Integration tool.



(3)

## Supported File Types by File Layout

a) CSV 101, Ravi, 10000

b) XML <Tag>101<Tag>Ravi<Tag>

c) Fixed Length 101...|Ravi.....|  
5 10

## Properties:

- 1) File Layout Definition
- 2) " " Segment / Record
- 3) " " Field

Eq:

①

EmpId	Name	Sal
101	Ravi,S	100
102	Giri,T	200
103	Raj,A	300
104	Rahul,B	400
105	Shantanu,M	500

→ Fixed Length file

c:\temp\ "BA4FILELayout.txt"

(2) Create a Table (B24-File-Tbl)

Emplid K  
Name

[Proj:- FileLayoutPro]

Salary (comprate)

(3) Create a file layout

a) Go to File

↳ New → file layout:

b) To insert record into ..

Insert → Record.

↳ B24-File-Tbl.

Insert  
↓

c) To provide file loc.

Go to Preview Tab

Default file Name

Browse  
↓

d) File Layout properties.

Format :

Save B24-Flout.

e) Preview Tab.

Segment Name :

See the data at last

If data is not populated properly  
In that same window, select the particular  
field & provide the length.

click on Save

(8) Execute the file layout prgm.

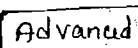
For it create a AE prgm.

↳ Just click on  Button

provide AE Name - B25\_Flout-AE

[To run file layout prgm, AE prgm is required.]

Now execute AE prgm.



↳ Restart

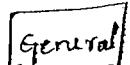
click on  icon.

(9) Now execute the tbl in SQL Server

File Layout Definition properties:

Open file layout

↳ click on prop. icon

 Tab -

Same as Before

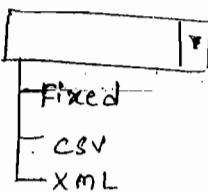
gets  
highlight  
for  
cl

### Use Tab

File layout Name : Bas\_flout

" " Type : Fixed

File " Format :



Definition Qualifier :  gets highlighted for CSV  
Delimiter :  comma, semicolon, other.  
eg: 

101	RAVI,S	100
①	②	③

 → For this we select Def. Qualifier  
101, "RAVI,S", 100  
①      ②      ③

If Data consists of a separator then it is used. (②)

File Definition Tag :  gets highlighted for XML

Buffer size :

Used depending upon the file we used.

Check the size of file we are using & ↑ the Buffer size.

Imply Decimal place

↳ apply " pos. to no. field "

Excel format

Whenever we are working with excel file select this.

Qualifier optional

101,"Ravi,s",100

If we select this we can place Qualifier which has separator value.

convert tabs to spaces

convert one tab to 8 individual spaces

Interpret Backslash

101\s Ravi,s\s 100\n 102\s Raj,k\s 200 ..

CSV — Format

Space — Delimiter

→ If we select this it will convert according to the backslash values.

strip whitespace

Remove unnecessary spaces before & after the value & upload the data

## Embedded strings interpretation

① Double quote    ② Backslash quote

101, "Ravi",  
      \ \$, 100.

If we place "" then sys treats " has a value and it stored as Ravi", \$

If we \ then sys treats " has a value and stored as Ravi", \$.

## File Layout Segment / Record:

Double click on Record Name in File Layout.

Use Tab

FL Name : B25-Float

Type : Fixed

File Record Name : B25-File-Tbl.

ID Seq. No.

↳ maintained by sys: we can't change it

Max. Rec Length :

↳ Depending upon length of fields

sys. came to know how many characters it has to store.

File Record ID :

To identify which row should go into which row

[001 101 Ravi,s	100	— Tbl 1
002 101 Ravi,s	200	— Tbl 2
001 102 Shiva,B	300	— Tbl 1
002 102 Shiva,B	400	— Tbl 2

Table 1      Table 2

=————— 002

↳ File record ID d

|    ID  
|    ↳ file start c  
|    pos.

3                  3  
|    ↳ ID length ]

Default qualifier

Field Delimiter

Record Tag

Record description:

### file layout Field properties

Double click on Field.

use Tab

Layout Name:      Seq. NO

Type: sm

↳ Seq. NO of field in  
table

Field Name

Trim spaces

↳ Removes unnecessary spaces.

Field Tag

↳ Label

Strip characters

[Eg: 101 Ravi, S 1000

Encrypting & sending data line

\$1# 0# 1\$ \* Ravi, \* S \* 1\* # \$0\$ 0

\$ # \* → These should not go to table.

So give these characters in strip characters]

Default value

Eg: 101 Ravi, S 100

102

→ Name is not populated.

Then give the default value for it.

Field Description

Field Inheritance

Record  } NO Need.  
field

field Pn

To maintain parent child relationship, make sure parent child should be first.

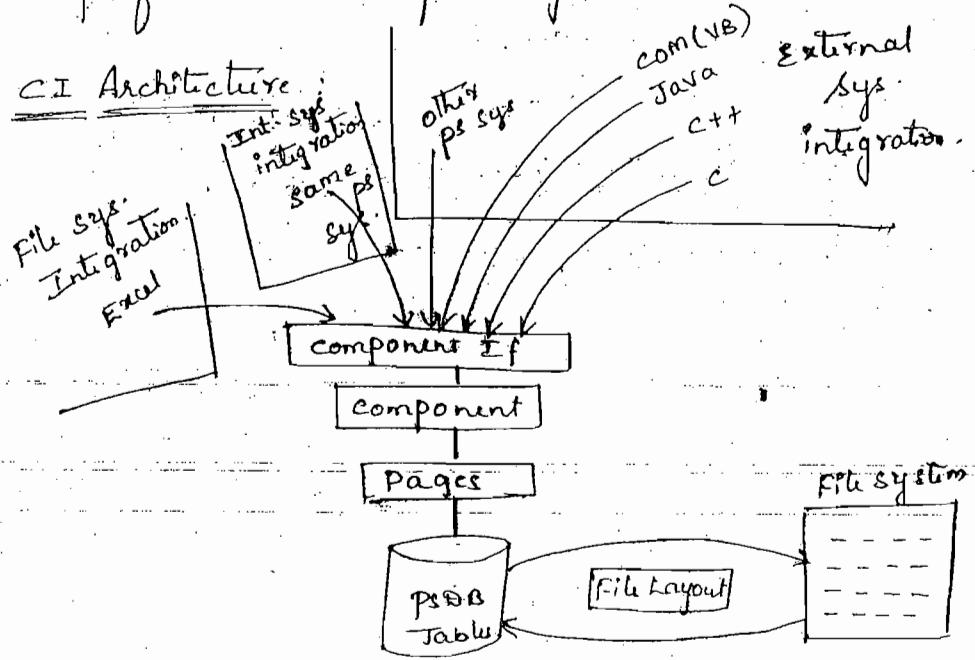
### Component Interface:

- proprietary tool
- Integration tool

### Advantages:

- used for Inbound operation.
- we can integrate with ext. sys.
- " " " other people soft sys.
- " " " in the ps system
- " " " Excel file (Data upload / Data migration)
- used when all business logics need to be performed while uploading data.

### CI Architecture:



Are you sure

## External System Integration:

### I Conditions:

① Java > 1.03 version

② Must have peoplesoft API in ext. sys.

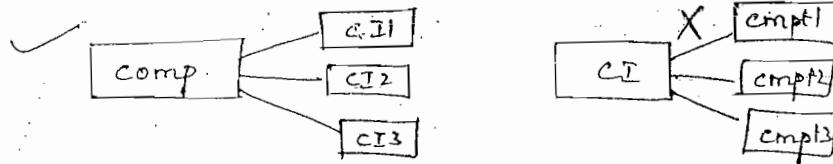
→ CI used for Bulk insertions. not only CI but any integration tool.

### Elements of CI:

① Name:

Used to differentiate one CI to other CI.

② Component:



Every component if it is based on a component

using one CI we can insert data into only  
one comp.

Get HIERARCHY ITEMS:

Edit Hierarchy Items

Get columns rows  
Interactive resp,

③ Keys:

④ Create keys:

Add Keys. Basically these are the fields which  
we see in Add New Value of component.

Ex



⑥ Get Keys :

Search keys, These are the keys we see on search page.

⑦ Find Keys :

- Search keys + Alt. Search keys, These are the keys we see on Advance search page.

Used for searching

Get Key Info Collection

② Returns collection of items that describes the get keys

④ Methods :

a) Create :

used to add new set of data to a comp.

b) Get :

performing basic search on comp.

c) Find :

used to perform advance search operation.

d) Cancel :

used to exit from component without saving.

e) Save :

used to save the component.

⑤ Properties :

These are the fields on the comp.

## Excel to CI integration : I

→ used to upload data from excel file to psdb using CI.

→ Required files:

a) Excel to CI.xls → used to provide data &

b) Excel to CI.vbs      insert data into psdb using CI.

    ↳ VB script file.

    ↳ Used to make a conn' to ps servers.

These files are stored at

PT 8.4 X → Excel

VL-CI-PROJECT

① Create a Record (B25-CI-DEST)

EmplId K, SR

Name ASK

DOJ Reasonable Date

comprati Required people code ( $>= 5000$ )

country prompt table: country-Tbl, Required

② Create a page (B25-CI-DEST-pg)

③ Create a component (B25-CI-DEST-cmpt)

④ Register the component.

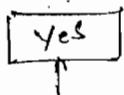
comprati → view people code

↓  
Field Edit (Event)

if B25-CI-DEST.comprati.Value < 5000 Then  
    Error ("Salary must be >= 5000");  
End-If;

### Creation of CI :

- ① File → New  
    ↳ CI
- ② This opens select source comp. for CI window  
    Name : B25-Cl-DEST-CMP†



- ③ Save B25-Empl-CI.

- ④ Build CI.

Build → peoplesoft APIs.



- ⑤ Add CI to permission list.

### Nav :

people Tools → Security → permissions & Roles

↓  
Permission List.

↓  
open any permission list.  
[AEAE 1000]

↓  
Go to CI page [+]

(B25-Empl-CI) Add CI →

### Steps

①

②

③

④

⑤

⑥

⑦

⑧

⑨

⑩

⑪

⑫

⑬

⑭

⑮

⑯

⑰

⑱

→ click on Edit



Select Full Access To methods



Steps for Excel to CI utility :

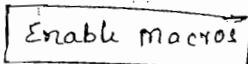
- ① copy the necessary files to desired loc.

Eg: Desktop - Folder

     copy ExceltocI.xls

ExceltocI.vbs

open ExceltocI.xls



Sheet Names in ExceltocI.xls

- a) Cover sheet:

This sheet specifies details about other sheets.

- b) Connection Information Sheet:

This sheet is used to make a conn' to the ps portal

- I) http://hrms:7010/psp/ps/?cmd=login

     ps login url:

     <protocol>: //<webserver m/c name>:<http port>/

        psp/<peoplesoft siteName>?cmd=login

web server m/c Name :

protocol :

Http port :

portal :

ps site Name :

Node :

hang code :

max. we can upload 65,256 lines bcoz it is  
Excel max. no.

chunking Factor :

↓  
specifies after how many rows of upload sys. must

issue a  
Commit to db.

For better performance use 40

Error Threshold :

↓  
specifies the no. of acceptable errors for a  
given chunking factor.

39 - 0

40 - 1

40 - 2

does not upload further rows

Action :

↓  
Used to specify in what action mode the comp. need  
to be opened.

### ③ Template sheet:

Used to create the reg. Template.

#### Steps to create Template:

click on New Template

Login :

User ID : ps
Password : ps
CI Name : B25_Empl-CI
OK

If we get error 'Non Authorized' it means we have not added it to permission list.

Highlight the fields which we are required &

click on Select Input cell:

To Deselect any cell click on

Deselect Input cell & Do Not include for sub.

click on New Data Input

Then we can see the fields in Data I/p sheet.

### ④ Data Input Sheet:

This is used to provide data that need to be inserted into our ps sys.

Enter the data in Data I/p sheet

101 Ravi, S 11/1/08 9500 IND  
:

After providing data click on

Stage Data for submission

moves data to staging & submission sheet

click on Submit Data

② Staging & Submission sheet:

This is used to submit data to psDB from Excel sheet as well as this is used to view Error or success msg.

when we click on Submit Data it provides

the status of the row in staging & sub. sheet

previously=we cannot see status column in Data I/p sheet.

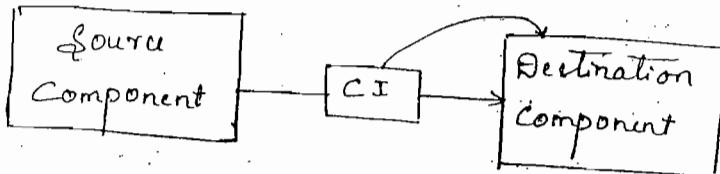
Go to Staging & Submission page

click on post results

This inserts status column in Data I/p sheet.

Now do the changes acc. to the status provided.

### Component to Component Integration:



Source page

Destination Page

Source Record

Destination Record

→ CI builds on Dest. comp.

Create the source record, source page & source comp.

Create source record with same structure as before table.

Take the prev. created record, page, comp as destinations, use the same comp if...

→ Go to source record.

↓  
on any field (select any field).

View people code.

↓  
Write in save pre change.

Drag & Drop CI on to it.

Do the changes as follows:

\* \* Set CI Get/ create keys \* \*

i) &OB25Emplci. Emplid = B25-CI-Source. Emplid .Value;

ii) Execute Get Method & put comments

iii) Remove comments for create method & ; after Then

iv) For each & every field do the mapping by removing remarks in Get/ Set CI properties.

Eg: &OB25Emplci. Emplid = B25-CI-Source. Emplid .Value

v) Remove comments for Save method

Save the record & project.

[Give the data in the source page, those values gets populated into Dest. comp.]

→ To see the coding of buttons (Input cell, Deselect, Staging & sub-buttons...)

Open ExcelCI

Tools → Macros

↳ visual Basic

## Configuration Manager:

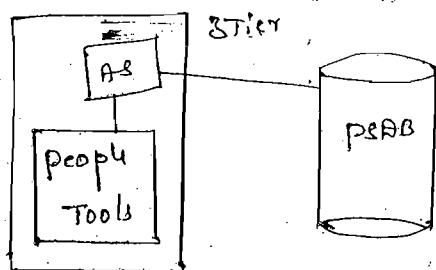
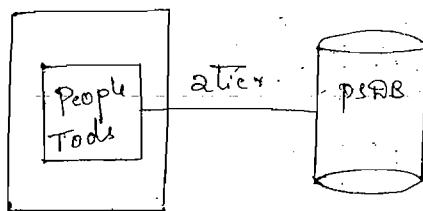
- This is a proprietary tool.
- This is a administration tool.
- This is used to maintain conn' b/w peopletools with app server & DB.

### Nav:

Start → program files → peoplesoft < Text >  
 ↓  
 Configuration Manager.

This opens config.mgr window.

Used to →



startup Tab

Db Type : app Microsoft SQL Server

AS Name : Hrms

Ab Name : HR

User ID : PS

Connect ID : people → Default

Connect password : people

" " (confirm) : people

User can override

User ID

Ab Type

Ab Name

} If we deselect this user cannot  
change these values in App' Design  
login window.

Cache files

↳ Temp. memory location for people tools

Directory : c:\ps\cache

purge cache, Directories

Delete the unnecessary files in temp. memory  
location.

Display Tabs

↳ used to specify the size

crystal/bus. Interlinks Tab

crystal Exec path

↳ This is used when we are working with  
Crystal reports

### Client setup Tab.

This specifies what are the tools we can access with this sys.

### Edit profile Tab.

### Trace Tab

↳ this will be discussed in tracing of AR.

### Import/Export Tab.

Settings are done thro' this.

In company's generally config.mgr setting will not present. Then ask your team member or TL to send config file.

### Export to a file

↳ If he clicks on this it generates the config file. Ask him to mail.

then download that config file to your system & save it.

Then go to import/export tab & click on

### Import

then automatically your config.mgr gets configured with the values.

## Sign On process :

30/11/07

Eg:

① User ID / operator ID : Eg: PS, VPI....

Used to login to portal / end user window

② Connect ID :

Used to check the physical conn' b/o client & P&DB. Eg: people

③ Symbolic ID : Default is SA: Eg: Sal

Used to create a temporary conn' P&DB.

Table : PROPRDEFN — Operator Ref. Table

PSACCESSPRFL — Access profile Table

④ Access ID :

Used to create a permanent connection to

P&DB;

Eg: SA.

Ch

Ocr

C

C

C

C

C

C

C

30/11/08

## PeopleSoft Work Flow

5

Eg:

PS-B25-Leave-Tbl

oprid K

Emplid K [prompt table psrs-SCHR]

From-AT K

To-AT K

STATUS [APPR-STATUS] → Translate values

DESCR Pending, Denied,

Approved.

Page: B25-Leave-pg

constant: p

comp: B25-Leave-<empt>

Search Record: PS-B25-Leave-Tbl.

Register the comp.

## Creation of Work Flow:

① Create a worklist table

File → New → Record

BusprocName K

Activity Name K → EventName K.

WorklistName K

InstanceID K

TransactionID K

oprid

Emplid, From-AT,

To-AT, status, Reason

Save B25 - leave-WL Build it

② Create a Map

③ Create Activity

File → New

↳ Activity

Activity consists of 3 major objects

Step, Event, Worklist

Go to Insert

↳ Select Step, Event & worklist

& place it on page (Activity)

→ Step specifies from where we need to execute the workflow request

→ Event defines when this request executed.

→ Worklist specifies where this request must be stored.

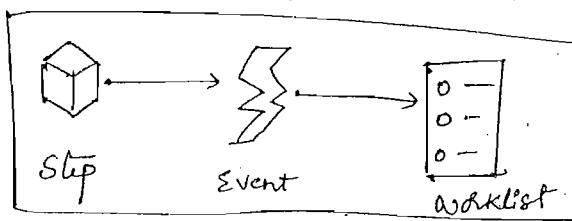
To map these 3 objects

Insert → symbolic link

click on step & drag it to Event

& again select another symbolic link

click on Event & drag it to Worklist



Double click on Step for step prop.

Step Ref:

Name: Step01

Attributes

Icon Descr: Leave Request

This opens a window

Menu Name: B24-Test-menu

procured by

page

If page name is not available, i.e. not

able to see, go to config.mgi.

click on purge cache Directory.

Bar Name: use

Page Name: B25-Leave-pg

Item Name: B25-Leave-empt

OK

Save the activity

↳ B25-Activity

Double click on Event

Name : event01

Icon Descr: leave req. Save

OK

(3)

Double click on worklist

Name : worklist01

Icon Descr: leave req. worklist

click on Attributes



Worklist Record : B25-leave-wL

worked by :

Business process Name : B25 - Bus process

Activity : B25-Activity

WL page Attributes

Mark worked when

Show Instances

User specified

Allow multiple unworked

Saved

WL acts as promptlist

Selected

Pooled list

Programmatic

Timeout processing Active

Time Out parameter

Reassignable

Late after

Days

Hours

Min

OK

(4)

### ③ Create Business process

File → New

↳ Business process



Consists of Activity

Drag & Drop the activity

Save → B25 - Bus process

### ④ Double click on Event (Activity)

Record Name : B25 - leave - Tbl

Edit Business Rules



This opens people code window in Backend.

Oprid (field) ▾ workflow ▾

if B25 - leave - Tbl . appr - status . value = "P" then  
 event  
 End - If ;

To execute workflow we have Trigger Business event.

Go to people books → people code.

↓  
open trigger business event

Copy the eg. & paste it in place of

Event

If B25\_Leave-Tbl. Appr-status.value = "P" Then

& Success = Trigger BusinessEvent (Busprocess, "B25\_BusProc"  
BusActivity, "B25\_Activity", BusEvent, "Step01")  
event

End-if;

- ⑤ Map the fields b/w actual reg. table & worklist  
tbl.

Double click on Worklist

click on **Field Mappings**

Message map

Worklist table fields	oprid	copy	B25_Leave-Tbl.EMPLID	From - RT	Actual table fields
	EMPLID	copy	"	"	
	From - RT	copy	"	TO - RT	
	DO - RT	copy	"	"	
	DESCR	copy	"	DESCR	
	STATUS	copy	"	STATUS	

copying data from actual reg. table page to  
worklist table.

For oprid → map with the manager

For this we need to write the query

These Queries are called Role queries

Field Name : oprid

Role Name : Emp. Review. Manager



Go to Frontend, Enter the data in the component.

When we click on save it saves in the worklist table.

→ Manager goes to worklist Tab in the frontend, there he can find all the pending requests.

When click on that particular request, it takes to the same page & he set the status as Approved & say OK.

Email Notification

Go to Activity → Insert

↳ Email

Select symbolic link & connect b/w event & click on Field Mappings

& enter the required details.

Eg. for file layout using parent child relationship

[proj :- Samplefile] with record id's

Parent record (Dept1)

DeptId K

Dname

city

child record (Empl1)

DeptId K

EmplId

Ename

Address1

Sample.txt

001, 10, HR, GN.

001, 20, PI, GN

002, 10, 101, AA, ST1

} Should enter into table1 (Dept1)

002, 10, 102, CC, ST2 } " " " table2 (Dept2)

Create file layout (File Samp)

Insert → Records

↳ Dept1, Empl1

Establish parent child relationship.

b/w these tables.

File Samp

↳ Dept1:

DeptId, Dname, city

→ Empl1

DeptId, EmplId, Ename, Address1

Adjust the  $\leftarrow$   $\rightarrow$   $\downarrow$   $\uparrow$  for the parent child relationship.

click on  $\boxed{AE}$  button.

It generates the AE prgm which is used for the execution of file layout.

Run the AE prgm with run control id provided.

This populates the data in tables. Observe in

SQL Query Analyzer.

11/12/08

## Tree Manager

### Navigation:

Tree Manager → Tree Manager

→ Only in the portal environment

### Definitions:

① Tree structure

② Tree

First we have to create tree structure, based on structure we have "tree".

∴ structure is a prerequisite for tree.

→ Tree represents database field values in a hierarchy.

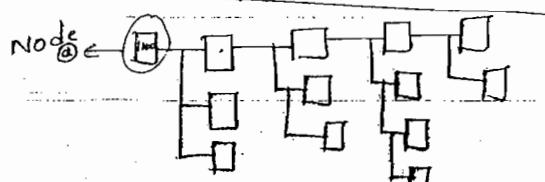
↳ key field:

Eg: Employee

EmplId	.....
1100	
1000	
1101	
=	
Σ	

cannot find any hierarchy.

levels → P VP M S C



Each one represents  
an employee.

→ Tree manager is a peoplesoft reporting tool.

## Tree types :

### ① Node oriented Tree :

Mainly used in HCMs for row level security.

### ② Detail Tree : used in following

Reporting Tools .

↑  
④ psnvision  
⑤ cube Manager.  
⑥ Summary Tree.

If we want to create  
these, the source is  
Detail tree .

⑦ Query — As an expression.

⑧ General ledger

### ⑨ Dynamic Detail Tree : → uses 2 db fields .

Used in psnvision

One is under Nodes Tab  
Other " " Details "

### ⑩ Summary Tree : Both fields must belong to same set ID .

Used in psnvision .

→ psnvision reporting tool is mainly used for  
General ledger module .

## Tree structure Types :

### ① Detail structure

### ② Summary structure

→ Node Oriented Tree

Detail Tree

Dynamic Detail Tree

Summary Tree → The structure we use is Summary

→ If the field is non key field we cannot create a tree for it.

Tree Elements :

Mainly consists of 3 elements.

① levels :

3 Types of levels

- a) Not used
- b) Strictly Enforced
- c) Loosely Enforced

② Nodes :

2 types of Nodes

- a) Node represents db field value
- b) Node represents logical value.

③ Details :

- a) Details represents db field value (or) a range of db field values

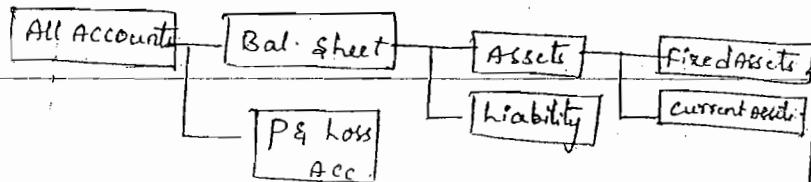
### Strictly Enforced:

child Node should be placed in the immediate next level. we cannot place any other level.

### Loosely Enforced:

we can place any level after the parent level.

Eg. for Node represents logical value



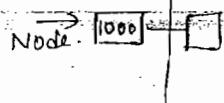
These are called logical values.

Detail Tree →



Eg. for Node represents db field values:

levels → P | vp m s c



→ Node oriented

Tree

### Icons:



— Node (collapsed)



— Node (Expanded)



— Node (Branch Node)

 — Node (Root node for Branch tree)

 — Detail.

→ In the tree structure, we have to provide the address where the tree components should be stored while creating tree.

→ While creating Tree structure we have 4 Tabs.

#### Structure

Tab

→ Type of structure.

→ This should be mentioned for control Table

→ " " " " " for Tx table.

Tab

Record Name :

Page Name :

Comp. Name :  Menu Name :

Tab

Field value :

Record Name :

Page Name : [ ]

Comp. Name : [ ]

Menu Name : [ ]

### Details Tab

Field Name : [ ]

Record Name : [ ]

Page Name : [ ]

Comp. Name : [ ]

Menu Name : [ ]

For creating Tree structure

Tree manager

Node oriented

Create New Tree structure

Structure

Tree structure ID : B25 - Deptid.

Description : Dept Node oriented

Type : Detail [ ]

Add key field

Objid Indirection

BU

User defined

None

Navigation options

Node multi-Navigation

Detail " "

### levels

Structure ID : B25\_DeptID

Record Name : Tree\_Level\_Tbl } These are the  
Page Name : Tree\_Level } defaults provided  
Component Name : by ps.

Menu Name :

Menu Bar Name :

Menu Item Name :

### Nodes

Default values are provided here, which are valid for logical values

Structure ID : B25\_Deptid

Record Name : Dept\_Tbl

Field Name : Deptid → Here it displays

Page Name : Department only key fields

Component Name : Department

Menu Name : Design - chart fields

Menu Bar Name : use

Menu Item Name : Department

### Save

6.

### Details Tab

keep it blank, we are not creating detail tree.

~~26/12/08~~ Tree manager → Tree mgr — For creating tree.  
create a New Tree.

create levels.

Set ID: Share

Add Level ↲

HQ, Functional

10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
888  
889  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
988  
989  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
999  
1000

25/12/08

## DataMover

- Comes under administration tools. (2-tier & 3-tier)  
→ Navigation: preferable.

- ① Start → programs → peoplesoft 8.x → Datamover (or)
- ② Go → Datamover [app' Designer]
- Uses:
- ① Main advt. is to copy data (DB objects) from one db to another db irrespective of RDBMS & OS.
- ② Data manipulation / DB updates. Datamover supports all SQL commands except Select.  
(Supports SQL Select stmt with subquery).
- ③ To create peoplesoft DB.
- ④ To take logical backups (DB backups).

Logical Backup - Suppose db is crashed then In order to restore db physical Backup is used.

→ 100% of db is restored & suppose if 8 to 4 tables are deleted i.e. some part of db objects are restored using logical backup.

Logical Backup will take more time.

Physical " " takes less time

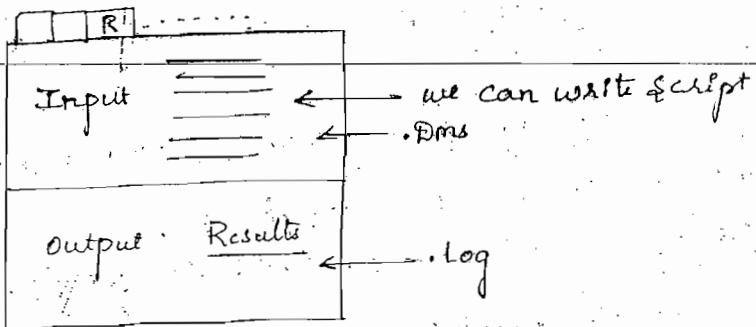
- ⑤ To apply patches / fixes & Tax updates.  
Majority of cases for this Datamover is used.

Using 'Change Assistant' tool we apply this.

advantage of Change Assistant is majority of steps is automated.

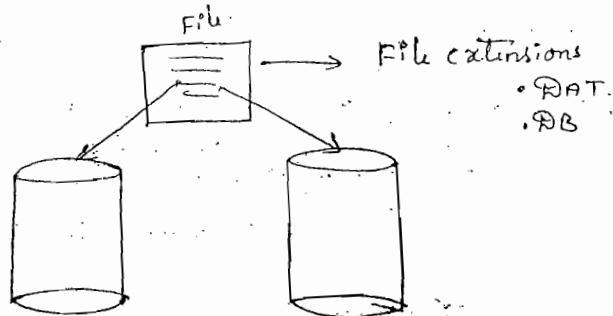
- ⑥ Rename records & fields. (Same we can also do from App' Designer)
- ⑦ Encrypt passwords.

DataMover seems like as follows.



- I/P  
→ file is saved with extension .DMS (Data mover scpt)
- when we click on Run button the script gets executed & the results will be seen in output.
- The output file is saved with .log extension.
- we cannot see 100% results in o/p window we have to see the 100% o/p in .log file.

→ we cannot move db objects from one db to another db directly. we have to use file as a mediator.



Entire db backup (logical backup) we use .DB extension.

Majority of cases we see .DAT

→ To write the script we have 3 types of commands

- ① Datamover commands
- ② SQL commands (DML, DDL & DCL except select)
- ③ Non-standard SQL commands

#### Datamover Commands :

- |                  |                       |
|------------------|-----------------------|
| ① Export         | ⑥ Rem/Remarks --      |
| ② Import         | ⑦ Rename              |
| ③ Replace - All  | ⑧ Run                 |
| ④ Replace - Data | ⑨ set                 |
| ⑤ Replace - view | ⑩ Encrypt - password. |

Encrypt - password : Encrypt one or all user passwords (operator & access) defined in PSOPRDEFN for users.

If we are entering info directly from Back-end

then we have to use this to encrypt the <sup>all users</sup> <sub>pw's</sub> password. Syntax : Encrypt - password { user ID ! \* } ;

Export: Select rec. info & data from records & store the result set in a file. You can use the generated export file as input for import command for migrating to another platform or within the platform.

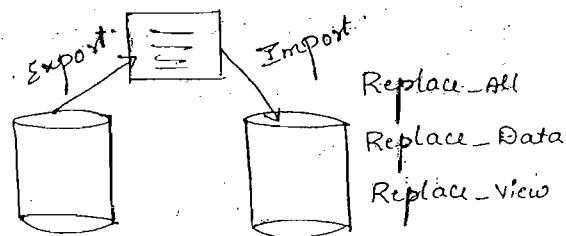
Syntax:

Export { record | \* } [ where condition(s) ] ;

→ Records exported using Export should have maximum of 500 columns.

Eg: Export ps-JOB; → Specified rec. must be a table, not view.

Export \*; → Export all ps records, including views.



Exports ① structure of table & the data in the table

- ② Index on the table
- ③ Views on the table
- ④ Triggers on the table.

Imports ① structure of table & the data in the table.

- ② Index on the table
- ③ Views on the table
- ④ Triggers on the table.

\$etNOviews → is only valid with Export \*

Import: creates db space, create non-existing records & Indexes & appends non-duplicate rows of to records. In addition, creates views if the export file was created using Export \* & imported using Import \*.

Syntax:

Import {record / \*} [IGNORE\_DUPS]  
[AS new-table-name];

while writing Import & export commands we have to give the filename.

If we don't specify the filename it uses the default file Datamove.Dat.

But recommended to give a filename.

Eg: Set Import file-name;

import PS-JOB; } → To import single record from a file.

Replace-All:

Syntax:

Replace-All {record / \*} ;  
[AS new-table-name];

If we use Replace-All, suppose 'JOB' table already exists then it drops the 'JOB' table & copied the data i.e replaced with the record (JOB) in the file.

→ Records defined using Replace - All can have a max. of 500 columns.

### Rem/Remarks / -- :

Used for writing comments.

Single line comments : -- This is my comment.  
--   

Multi-line comments : Remark

anyone must use ..

### Rename:

We can rename records & fields.

### Syntax:

Rename Record oldrecname As Newrecname

Rename Field oldfieldname As Newfieldname

Rename Field Recname.fldname As Newfldname

→ Renames the fld belongs to particular record.

→ Recommended to use this

→ Renames the field across the db.

After using Rename command, we have to run the alter command then only the change will effect in the db. (i.e. Table name & column) otherwise it will not effect.

### Run:

→ F

key

In

AB

ETX

Set

Date

Par

( )

( )

( )

( )

( )

( )

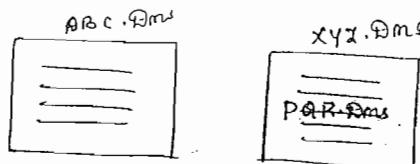
( )

( )

( )

( )

## Run:



XYZ.DMS

PAR.DMS

X

PAR.DMS

Run ABC.DMS

Here If we want to run  
the ABC.DMS script file to  
run then write

Run ABC.DMS as follows

→ Run supports only one level.

## Syntax:

Run Scriptfilename.

In XYZ.DMS it executes PAR.DMS; Again it contains ABC.DMS. it will not allow to execute. It displays error msg. supports only one level.

Set: Useful for setting the environment to run the Datamover script smoothly.

## parameters:

Set output

Set NORecord

Set Input

Set NoData

Set Log

Set Noviews

Set Notrace

Set NoIndex

Set Ignore\_Dups

Set No commit

Set Commit NNN

Set start

Set startafter

Eg: ① set output ABC.DAT

Export ;

② set input ABC.DAT

Import ;

③ set log ABC.log

④ Two Tier

Config. Manager → we can enable Traces

The traces enabled in config.mgr is applicable  
to all tools working in 2-tier.

If we use 'Set NOTrace' while running the  
script, Datamover disables the traces in configmgr  
& after execution it automatically enables the  
traces.

⑤ set NoRecord — Rec. Rec. Def not exported/imported

⑥ " NOIndex — Indexes " "

⑦ " NOData — Data " "

⑧ " NOView — Views " "

⑨ set Nocommit — Sys. will not commit until  
the script gets executed completely

⑩ set commit 10,000 — Commit for every 10000 rows.

⑪ Suppose 8000 records if error occurs at  
2<sup>nd</sup> record. ABC — error (1<sup>st</sup>)

ABD — error (2<sup>nd</sup>)

Set Start ABC — From ABC it starts

Set startafter ABC — From ABD it starts

#### Replace - Data:

This command is a variation of the import command. Use it to delete data in existing tables & inserts the corresponding data from the export file.

#### Syntax:

Replace - Data { record | \* } ;

#### Replace - View:

Recreates one or all specified views in the database.

#### Syntax:

Replace - View { View | \* } ;

→ If you use Replace - view in bootstrap mode, the sys automatically activates 'Set Ignore Errors'. This enables peoplesoft datamover to continue processing until all of the views definitions have been processed, & all errors have been written to the current log file.

## NON-standard SQL Commands

- ① Store      } used to handle  
② Erase      } COBOL SQL statements

→ Use the commands to change COBOL SQL statements in:

ps-SQLSTMT-TBL

Store — Insert, Erase — Delete

### Syntax:

Store programname-type-stmtname

### Syntax Rules to be followed while writing script:

- ① No case-sensitive.

For this rule the exception is Quoted literals.

Eg: Export emp-rec where location = 'Hyd'  
case-sensitive → ('hyd', 'HYD')

- ② we can use either record name (or) Table name

Eg: Export JOB ; (or) Export ps-JOB ;

For this rule the exception is use in criteria

we have to specify tablename not recname

- ③ No problem with empty lines & white spaces

Eg: Export ps-JOB ; (or)

Export ps-JOB ;  
[Empty Line]

Set NO views;

④ Each command should be ended with valid separator.

Valid separators are ; (or) /

For this rule exception is single line comments.

⑤ We can place separator at the end of each command line (or) ~~immediately~~ in the 1<sup>st</sup> col. of next line.

Eg: Export ps-JOB;

(or)  
Export ps-JOB

⑥ Run will support in only one level.

Modes of DataMover:

① Regular Mode.

② Bootstrap mode.

Bootstrap mode:

Navigation :

Start → programs → peoplesoft 8.x → Datamover

→ Signon with AccessID & password: (DB user)

DB user & peoplesoft user are diff.

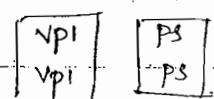
[sa]  
[sa]

Regular Mode:

Navigation :

① Start → prgms → peoplesoft 8.x → Datamover

→ Signon with peoplesoft user ID & password



② Go → DataMover [App' Designer]

Eg: ①

→ Bootstrap mode limitations: (Some commands are not valid)

① Export

② Rename

③ Replace-View. → Some situations this is valid.

→ Bootstrap mode is used for importing purpose mainly.

→ Regular mode does not have any limitations.

→ To install PSOB we will go for Bootstrap mode.

→ Eg: set output ABC.DAT;  
Export ps-JOB; ↗ where to create?

① Specify the path C:\Temp\ABC.DAT;

② Otherwise it looks for the default file paths in

Config. Mgr. → [Common] Tab

→ Oracle

sqlserver

SQL+

Query Analyser

SQL\*Loader

Export & Import

→ peopleSoft developed DataMover irrespective of RDBMS.

If we don't have idea of SQL Server (or) DBA.

we can still work with DataMover & the perform the operations as SQLServer.

Eg: Open Data mover.

```
set NO Trace;  
set output Batch25.DAT;  
Export TRNL-Header;  
Export TRNL-LN;  
Export product-TBL where setJP = 'Share'  
Save → Finance - P78.4x → Scripts  
↓  
Batch25.DMS
```

See the .DAT file → specified in config.mgr→path.

→ Indexes in peoplesoft.

while building the record, 2 indexes will be created.

① Non-clustered index — 0 to 9 indexes it will create  
max. 9 indexes; created based on alt. search key

② primary index.

created based on duplicate keys & keys.

only 1 index for 1 record.

open a Record → Tools → Indexes

→ Max. we can have 10 indexes.

→ 'Store' command first delete the existing stored stmt  
from PS-SQLSTM-TBL & then inserts the new stmt  
using the syntax.

→ Erase command deletes one or all stored stmts  
from PS-SQLSTM-TBL.

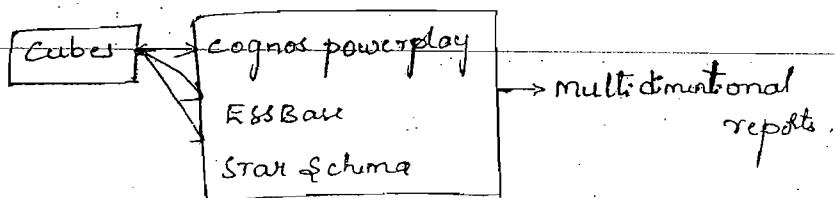
26/12/08

## PSNvision

- This is a reporting tool.
- ps app's reporting tools
  - ① SQR
  - ② crystal
  - ③ Nvision
  - ④ pequery
  - ⑤ cube manager (using as OLAP tool)
  - ⑥ XML publisher (Available from 8.48)

which

- Using cube Manager we create cubes, will becomes source for multidimensional reports.



- Nvision is used for finance General Ledger module.
- crystal reports are mainly on control Table data.
- SQR, " ", " ", " ", Tx, " ", "
- PSN vision is a proprietary Tool (people tool).
- " " mainly based on Microsoft which is a 3<sup>rd</sup> party tool.

Steps to be followed for psnvision Report:

- ① Define the nvision layout. This will be defined on Excel.
- ② Define the scope definition (optional)
- ③ Define the nvision report request.

## ④ Run the Report Request.

→ While creating Revision Layout we have 2 types

① Tabular → Source is Query (PS Query).

② Matrix. Reports created are

    ① Source is Query, tabular reports.

    ② " " Ledger → Reports created are crosstabular reports.

→ Tabular reports are called as Detail reports.

→ Crosstabular " " " " " Summary " "

→ If we are using Query in matrix layout, that query must contain aggregate func.

→ In Tabular layout, query may be anything.

### Navigations:

① Start → Programs → PeopleSoft 8.x → penvision.  
(or)

② Go → penvision [App' Design] → [Windows]  
(or)

③ Reporting Tools → penvision [portals].

→ Defining the Revision Layout is only done in the development env. i.e. in windows.

we cannot define layout in portal.

→ Other 8 steps, scope def., report request,

running the request can be done in any env. (i.e. may be in windows (or) portal)

Additional features of prcision in Portal.

- we can able to create Report Book.
- If we want to execute the report from process schedule, we can do it.
- we can schedule the time for report.

Eg! Tabular layout

① creating a layout

i) Menu → prcision → new layout

[Recommended is first check whether any layout is suited for our requirement or not. If available Save as layout, and make the changes as per our req.]

• XNV is the extension. (B26-Tabular)

ii) Menu → prcision → Layout Def

① Tabular layout sheet

② Matrix " "

Prompting options

Setid :	Share
Effid :	
BU :	US001

OK ↵

### PS Revision Layout Def.

**Summary Tab**

First highlight the excel sheet & click on

**Source Tab**

Type **Query**

**QueryName**

Select the Query we have req.

**Option Tab**

**OK**

Go to excel sheet & select column.

Again go to revision → layoutdef.

Here we can able to see another Tab named

**Column Tab**

click on it & select the req. field's.

click on **OK** after selecting the fields.

Thus Layout Definition is designed.

**(2) Scope (optional)**

**② Report Request**

Reg. Name : **B26-Tab**

Report Title : **Sales By Product**

Requesting BO

[U3001]

Layout

[B26-Tabular]

o/p options

Type : [File]

Format : [Microsoft Excel files]

[Save ↴]

[Run ↴]

Matrix layout:

Terminology in General ledger:

① Chart of Account (chart Fields)

↳ Accounting structure

In what way we are creating the accounts

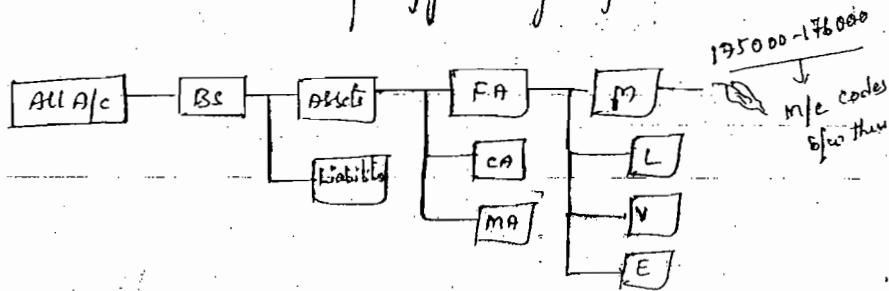
② Calender

↳ Represents periods

③ Time span

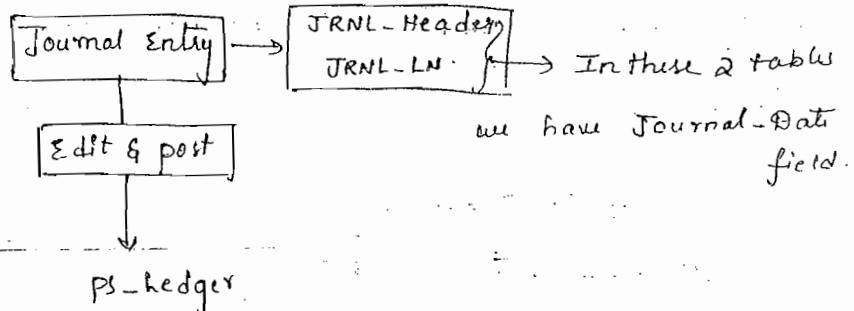
④ Ledger

⑤ Trees → used to specify range of values



ledgers:

Majority of ledger details are stored in ps-ledgers



BO	Ledger	Acc.	Dept	product	...	Fyear	period	Amt.

In ps-ledgers we don't have any Date field, but if we want the report from particular date to particular date we have Fyear & period fields.

From year

To year

From period

To period

calender

we can have many calenders, In order to specify periods we need to give calender name.

Eg: 998 → Adjustment period.

Suppose we are dealing with monthly period,

while recording this year Tx, we find prev. year Tx, in such cases we cannot use it in this year Tx, for recording prev. year Tx we use 998 adj. period.

→ O No adjustment period

O Adjustment period 998 -

O " " From  To

998 → entire year

901 to 912 → adjustment period monthly wise.

### Timespan

From year

To year

From period

To period

Calendar

O No adjustment period

O Adj. period

O " " From

OK

Include Balance Forward

→ while generating reports we have to specify the chart field name, (Account, Dept, pdut, -)

v. year → special periods in ps.

this year opening period - represents zero

closing period - " 999.

All opening Balances goes to zero balance.

closing Balances of prev. year will become the opening " for this year.

→ calendars/schedules → Timespan.

Time spans:

Start year:  start year Type: Relative to current

year  
(or)

Absolute year.

If we specify Absolute year & given year as 2008.

It always give the balance sheet of 2008. If we want to get the prev. years balance sheet reports we have to specify Relative to current year.

If year: , Type:

It displays the bal. sheet of 2007.

Starting period is always 1.

If we want Bal. sheet as of date then specify the type of period as Relative. End period: zero.

start period:  Type:

End period:  Type:

Eg: NVision → New Layout

b26-qmat

Matrix Layout Sheet

SetID :

Eff Date :

BU :

NVision → Layout Def

Source Tab

Type : Query

QueryName :  Journal-NVision - 'Journals'

This query must contain aggregate func.

If we select the Query & click on  It displays the aggregate func. list columns ..

Time spans

click on any column on Excel Sheet & click on

Filter Tab

(Before clicking select any row on Excel sheet)

Select the chart field

Field

0 Dimension

Account

Local ledger

Filter options

0 Selected Tree nodes

0

0

0

Tree / Hierarchy : ACCTROLLUP

we have to select the Tree level : summary

Tree Node : Capital

Select Capital

Retain contents

Apply

confirm changes

Delete capital Select the nodes required for  
the report. (Liabilities, equip).

If we run the request, Labels will not be  
displayed.

Go to revision  $\rightarrow$  Layout def.

Labels tab.

Enter Node

Descr

## Matrix (ledger based):

Nvision → New Layout

B&G-LMAT.XNV

Nvision → Layout Def.

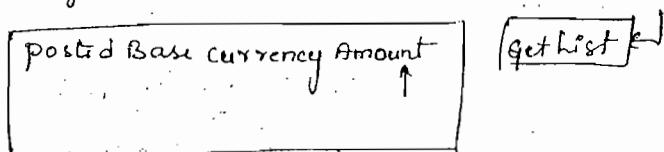
Select O matrix

Source Tab

Type : Ledger

Ledger Name : Local currency ledger (Local)

Ledger Amount column



Apply Select A1 in Excel sheet

Select any cell & double click on it

(If we hardcord 31/12/2008 it used only for 2008 year)

Go to variable Tab

① Date & Time periods

This displays of. ASD %. in that particular cell.

Select on any cell double click

② Report Request

Select OPRID

of. OPR %

of. RBU %

Scope Related

% RIO %

% LYN %

→ PeopleTools → utilities → administration → stringTables.  
To define the strings.

Equity share holder funds → If we type this in excel sheet, in future if it changes we have to change it so in order to use this from vision we have to define the strings : create

Add New value      B26\_LMAT Table.

String ID

LTL

Longtermliabilities

STL

Short "

CTL

current "

If in future it changes longterm liabilities to long term values instead of going to devt. env. & change go to portal & change the values.

FA

FixedAssets

CA

Current

MA

Misc

→ we can have 'n' no. of string values.

Go to strings Tab

Program

B26\_LMAT

Stringid

← Select the req. string

Use Navigator to move to next fields.

To print the values

Highlight a column in Excel.

Timespan

Go to  Tab.

Select Tree, nodename

% LTL %

% FA %

% STL %

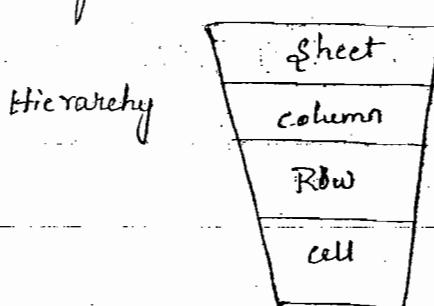
% CA %

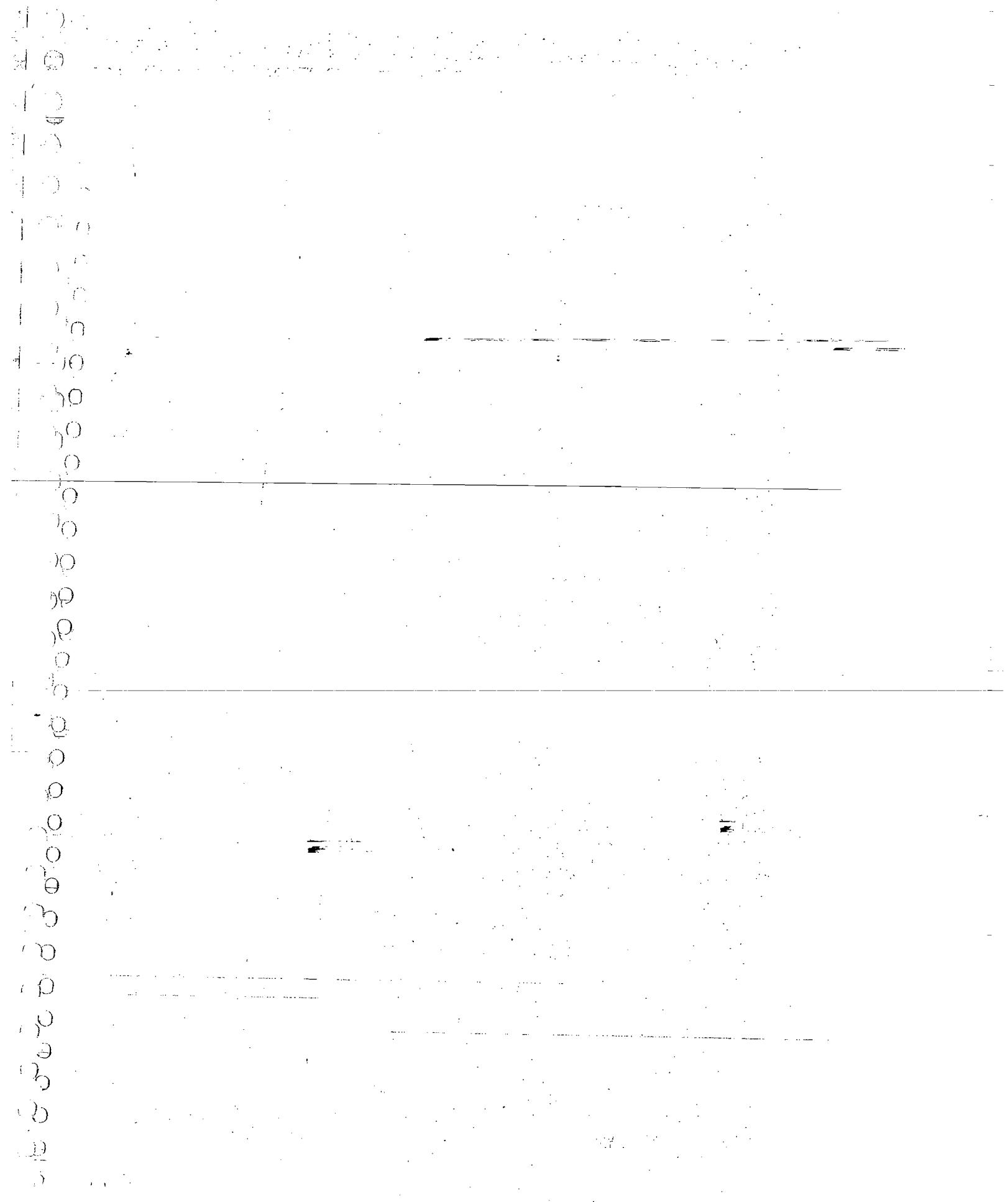
% CTL %

% MA %

It will print the values same for both liabilities & assets b'coz we selected full row and applied the tree node values.

In order to get the assets values click on the req. cell & go to  Tab. Select the Node values as before.





1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100

8/11/08

## Finance

75

→ PS supports diff. app's

HRMS / HCM

Finance / SCM

EPM

CRM

Student Admin.

→ modules in Finance are

- 1) General ledger
- 2) Account payables
- 3) Account Receivables
- 4) Asset Management
- 5) purchasing / purchase order
- 6) Billing
- 7) Inventory
- 8) Project costing
- 9) Order Management

General ledger:

Heart of Financial system.

place where all your data will be reconciled

checked

Account payables:

place where any kind of payments take place.

Account Receivable:

place where cash is received.

Asset Management:

place where assets are maintained.  
↓

Items which will have resale value.

Purchasing/purchase order:

place where orders are placed to purchase items.

Billing:

place where we generate bills.

Inventory:

place where items are maintained.

Project costing:

place where projects are maintained.

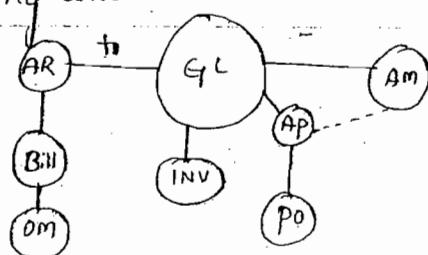
Order Management:

place where orders can be received.

→ General ledger have 2 kinds of flow

1) -ve flow : spending amount

2) +ve flow : Receiving amount.



Purch

Purch

1) Re

2) A

3) Bu

4) Re

5)

6) Cl

7) A

8) G

9) E

10) L

11) G

12) E

13) L

## Purchase Order : / Purchasing :

### Purchasing Roles :

#### 1) Requestor :

Is a person who requests for goods

#### 2) Approver :

Person who approves the request

#### 3) Buyer :

Person who buys the items / goods.

#### 4) Receiver :

Person who receives the goods & inspects the goods.

### ① Business process :



Step by step process.

1) Create Requisition

2) Approve Requisition

3) Get Quotations

4) Create purchase order

5) Approve purchase order

6) Dispatch purchase order

7) Create Receipt

8) Perform Inspection process

9) Return to Vendor (RTV)



i) Create Requisition :

→ Requisition is nothing but a form which consists of details about request for goods.

→ Requestor creates Requisition.

Details :

1. Requestor.

2. Item List.

3. Quantity.

4. Unit of measurement (UoM)

5. Vendor details.

6. Delivery Date.

7. Buyer Details.

8. Ship to Location : place where goods will be delivered.

10. Price.

9. Distribution Details.

→ Every Requisition will have a requisition id.

Navigation :

purchasing → Requisitions → Add / update Requisition.

Records :

REQ-HDR → Requisition Header Table.

REQ-LINE → " Line "

RE

a)

→

→

→

→

→

→

b)

P

O

O

O

O

O

O

O

O

O

O

O

REQ-LIN-DISTRIB → Requisition Distribution Line  
Table.

2) Approve Requisition:

- place where requisition will be approved.
- place where item, quantity & amount will be approved.
- Approving can be done by levels
- These levels can be defined using quantity or price.

Types of Approvals

a) complete Approval

- ✓ 10 - comp - 100 - 1000
  - ✓ 10 - chair - 20 - 200
  - ✓ 10 - Table - 50 - 500
- } Requisition Lines

where all requisition lines & quantity are approved.

b) Partial Approval

- ✓ 10 - comp - 100 - 1000
- ✓ 10 - chair - 20 - 200
- ✗ 10 - Table - 50 - 500

where all lines of entire quantity may not be approved.

### Navigation:

purchasing → Requisition → Approve Amounts  
(or)

worklist → work list

### Records:

REQ-HDR → REQ-STATUS (this value will change  
from opened to approved (or)  
" " denied)

REQ-LINE → REQ-STATUS.

### 3. Get Quotations:

is outside of ps system, they get quotations  
from diff. vendors & then they decide which  
one to buy.

Dell — 1 — 100

HP — 1 — 80

Sony — 1 — 110

### 4. Create Purchase Order:

#### Types:

##### a) Requisition Based Po:

Creating Po from an approved requisition.

→ Depending upon the situation we decide whether  
we need Requisition or not.

## b) Non-Requisition Based Po:

Creating Po without a requisition

Eg: Stationary

### Navigation:

Purchasing → purchase order → Add/Update Po.

→ created by Buyer

→ i) Po Date

ii) Item Details

iii) Quantity

iv) Price

v) Delivery date

vi) ship to location

vii) Buyer

viii) Vendor

ix) Distribution Details

x) UOM

→ Every purchase order will have a Po ID

### Records:

Po-HDR → Po Header Table

Po-LINE → Po Line

Po-LINE-DISTRIB → Po Distribution Line Table.

### 5) Approve Purchase order:

- place where quantity & price is approved.
- Doesn't support multilevel approvals.

Types:

#### a) complete approval:

All Lines & price is approved.

#### b) Partial Approval:

only some of the lines are approved.

Navigation:

purchasing → purchase order → approve Amounts

Records:

PO\_HDR → PO\_STATUS

PO\_LINE → PO\_STATUS

### 6) Dispatch Po:

- The process of sending Po details or Po info. to vendor.

Types:

#### a) Multiple Po Dispatch:

we can Dispatch more than one po

#### b) Single Po Dispatch:

we can Dispatch max. one po.

Disp

a)

b)

c)

d)

Disp

a)

b)

c)

Nay

F

C

D

F

E

E

→ We

O

### Dispatch Methods :

#### a) Print Po :

print po & send it to vendor via post.

#### b) Email :

Creates PDF file & mail to vendor.

#### c) FAX :

Creates o/p & fax to the vendor.

#### d) Phone :

Buyer places the order via phone.

### Dispatch options :

a) original Dispatch: Normal Dispatch.

b) Test Dispatch : Test whether Fax or Email is dispatching properly or working properly or not.

c) Duplicate order Dispatch.

### Navigation :

For Multiple Po Dispatch :

Purchasing → purchase order → Dispatch Po's

For Single po Dispatch :

Purchasing → purchase order → ADD/update Po.

→ we can only Dispatch approved po's.

(3) Program Type : SQR process (PO Dispatch process prgm Type)  
" Name : POPO 005. (PO Dispatch process program Name)

### 4) Create Receipt:

→ place where receiving details are captured.

#### Details:

- 1) PO ID
- 2) PO Quantity
- 3) Received Quantity
- 4) " Date

#### Types of Receiving:

##### a) complete Receiving:

$$\text{PO Quantity} = \text{Received Quantity}$$

##### b) Partial Receiving:

$$\text{PO Quantity} > \text{Received Quantity}$$

#### Another classification:

##### a) Normal Receiving:

we have PO Quantity

##### b) Blind Receiving:

cannot see PO Quantity

→ For single PO we can have multiple receipts

NAV

PLC

RP

1

2

→ 1

2

→ 1

3

TP

a)

b)

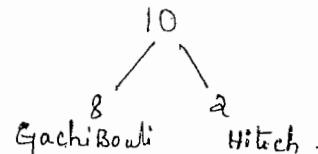
c)

d)

e)

f)

process prgm  
Type  
rocks  
time)



<u>Po</u>	<u>RQ</u>	<u>IP</u>	→ Inspection passed.
10	10	10	— complete
10	B	8	— partial
10	10	7	— partial 3 damaged.

Navigation:

purchasing → Receipts → ADD/Update Receipts

Records:

1. RECV-HDR → Receipt Header Table
2. RECV-LINE-DISTRIB → Distribution Line Table.

→ We can create receipts only for Dispatch Po's.

a) Return to vendor:

→ performed when received quantity is greater than inspection passed quantity.

Types:

a) Replace Goods:

	<u>PO</u>	<u>RQ</u>	<u>IP</u>	<u>cost</u>	<u>Paid</u>
1 <sup>st</sup>	→ 10	→ 10	→ 7	→ 1000	→ 700
2 <sup>nd</sup>	→ 3	→ 3	→ 3	→ 300	→ 300

b) Adjust Amount:

10 → 10 → 7 → 1000 → 700

## Account Payables :

### ① Vendor setup :

→ Vendor is a person who supplies goods / services.

Depending upon payment :

#### a) Single payment vendor :

Vendors to whom payment will be done only once.

#### b) Regular vendor :

Vendors to whom the payment is done on a regular basis.

Depending upon Tax/withholding :

#### a) with holding vendors :

1000 — vendor      800  
                        ↓  
                        200 (Income Tax)

Dell — 1000 — 1000  
|                    ↓  
800  
Income Tax — 200

→ Vendors to whom entire payment is not done  
→ where we withhold some amount and pay to income tax Authority.

b) Non-withholding Vendors:

Vendors to whom entire payment is done  
and he is responsible to pay taxes.

Depending upon service:

a) Supplier vendor:

Vendor who supplies goods/services.

b) HRMS vendor:

Vendors to whom amount is charged  
related to HR Activities.

c) Employee vendor:

Employee in a company.

charges like Reembursements, onsite travels etc.

d) Attorney vendor:

100

80 - vendor

20 - ITA → Income Tax Authority.

Vendors to whom we will be paying tax  
amounts.

Details:

- 1) Vendor ID
- 2) Vendor Name
- 3) Vendor Address

- 4.) Vendor phone Number
- 5.) Vendor email Address
- 6.) Vendor Fax Number
- 7.) Vendor Location
- 8.) vendor contact details
- 9.) Vendor payment details : (Payment mode  
eg: cash, check etc)
- 10.) vendor payment Terms : (monthly, yearly ...)
- 11.) vendor withholding Details.
- 12.) vendor Bank Information .(Acct/no)

### Navigation :

Vendors → vendor information → Add / update  
vendors.

### Records:

VENDOR

VENDOR\_PAY

VENDR-BANK\_ACCT

VENDOR\_ADDR

VENDOR-ENTCT — vendor contact table

VENDOR\_WTHD

VENDOR\_LOC

VENDOR\_PHONE

EFT - Electronic Fund Transfer

## ② Banks & Branches :

→ Used to specify the list of banks & branches where company has accounts.

### Navigation:

a) Banking → Banks & Branches → Bank Information

b) Banking → " " " → Branch "

## ③ Account Setup :

→ Used to specify the list of accounts used by company.

### a) External Accounts :

These are the accounts which physically exist.

### b) Internal Accounts :

These are the accounts which conceptually exists.

Used for reconciliation purpose

④ EA  
A company can have 2 accounts

a) Debit — Spending amount (-ve) Payments

b) Credit — getting amount (+ve) Receivings

### ⑤ IA

Reimbursement Account, Expense Accounts,  
Asset Account, Discount Account, ...

Reembursment Acc. }  
Expense Acc.      } Debit  
Asset Acc.        }

Receiving Acc. } credit.  
Discount Acc.     }

### - Navigation :

For Ext. Acc. :

Banking → Accounts → External Acc.

For Int. Acc. :

Banking → Accounts → Internal Acc.

### (4) Withholding Setup :

Tax/1099 Rule

Elements in withholding :

#### a) Withholding Rule :

plan where upper & lower thresholds, Tax %,  
additional charges are defined.

#### b) Withholding class :

Collection of Rules.

#### c) Withholding Jurisdiction :

Court to take legal actions for that location.

d) Attorney vendor:

Vendors to whom Tax amount is paid.

e) Withholding Entity:

place where we combine withholding classes

with jurisdiction & Attorney vendor.

Navigation:

→ For withholding rule:

Set Financials / supply chain → product Related →

procurement option → withholding → withholding Rule.

→ For withholding class:

Set Financials / supply chain → product Related →

procurement option → withholding → withholding class.

→ For withholding jurisdiction:

(Types & classes)

Set Financials / supply chain → product Related →

procurement option → withholding → withholding jurisdiction

→ For Attorney vendor:

jurisdiction

Vendors → vendor info → ADD / update vendors.

→ For withholding Entity:

Set Financials / supply chain → product Related →

procurement option → withholding → withholding Entity.

## Account Payables :

- used to perform any kind of payments.
- we can do payments only to vendors.

### (1) Business process :

- 1) Create a voucher.
- 2) Approve Voucher.
- 3) Perform Matching process.
- 4) Handle Match exceptions if any.
- 5) Post vouchers.
- 6) Perform payments
- 7) Post payments.
- 8) Close Voucher

GL

### 1) Create Voucher :

This is entry created in AP to do payments.

Depending on Po :

Records :

VOUCHER LINE

#### a) Po Based Vouchers :

Vouchers created out of an approved Po's

#### b) Non-Po based Vouchers :

Vouchers created without a po.

Depending on withholding:

a) Withholding Voucher:

Vouchers created for a withholding vendor.

In this vouchers we don't pay 100% of amount to vendor.

b) Non-withholding Voucher:

These are the vouchers created for a non withholding vendor.

In these vouchers 100% of amount will be paid to vendor.

Voucher Types:

a) Regular Voucher:

These are the normal vouchers what we create in AP System.

b) Prepaid Voucher:

These are the vouchers created to perform advance payments.

c) Adjustment Voucher:

These are the vouchers to adjust any amount to an existing regular voucher.

d) Journal Voucher:

These are the vouchers created to balance the

amounts & accounts in Acc. payables with GL.

e) Reversal voucher:

These are the vouchers created to make the amount to zero for an existing regular voucher.

→ Reg.

$$50 - 1 - 100 - 5000$$

PO

$$50 - 1 - 100 - 5000$$

Vchr  
Total  
8 open

prepaid voucher

$$25 - 1 - 100 - 2500 \rightarrow \text{DELL}$$

↓  
Prepayment/Advance amt.

Reg. Vchr:  $50 - 1 - 100 - 5000 \rightarrow \text{DELL}$

$$25 - 1 - 100 - 2500 \rightarrow 2500$$

Automatically advance amt is deducted.

Reg. Vchr → Dell

$$\text{Vchr} \rightarrow 10 - 1 - 100 - 1000$$

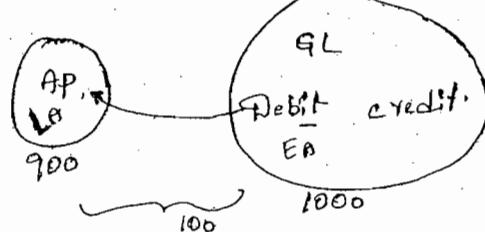
cheque

Debit
1000

For transportation - 100/-

Now we have to adjust  
to existing vchr

$$\rightarrow \text{AT vchr} - \text{Dell} - 100$$



100/- must be corrected in Logical Acc. (AP).

Always External Acc's value is correct.

Then we have to tally LA & made it to 1000.

For this purpose we use Journal Voucher.



Dell - 10 - 100 = 1000 → Reg. Vchr.

According to market price is 150.

Sys. paid 1000 but vchr is not passed.

Reversal vchr.

Dell - 10 - (-100) = (-100)

Already done the payment, but now we want to cancel the payment, such cases we use reversal vchr.

Invoice no: Bill number

Navigation:

Account payables → Vouchers → ADD/update

Details:

↓  
Regular Entry

1) Voucher ID

2) created Voucher

Freight amount: transportation charges.

3) Item Details

4) Invoice ID (Bill no.)

5) Invoice Date (when the bill has been created)

6) Payment Date (when we need the payment)

7) Vendor

8) Payment Method.

## 2) Approve Voucher :

place where approvers will approve the amounts.

Records: VOUCHER — APRR-STATUS

Navigation: VOUCHER-LINE — APRR-STATUS

Account payable → vouchers → Maintain Vouchers

Approve → Approve Amounts  
Voucher

## 3) Matching Process:

VOUCHER — MATCH-STATUS

Req.

Dell - 10	comp - 1 - 100	1000	Process type : App engine
PO			Process Name : AP_MATCH
Dell - 10	comp - 1 - 100	1000	
Voucher			
Dell - 10	comp - 1 - 100	1000	

PC  
E  
VPA  
C

4) M

a)

b)

Req.

PC

PO

VPA

C

5) P

d

e

f

g

h

i

j

k

l

Sys. checks whether quantity & price that are specified in requisition, PO & voucher during matching process. If these values are same " " goes to success if these values are diff. in any of the entries then sys. gives Match exceptions.

→ Until the match exceptions are corrected we cannot do payments.

PO 10 - comp - 1 - 100 - 1000

Dell - 10 - comp - 1 - 90 - 900

Voucher

Match  
exceptions

Dell - 10 - comp - 1 - 90 - 900

a) Match exceptions: AP → Batch processes

a) Override exceptions:  
↓  
= Vouchers  
↓  
Matching.  
System ignores match exceptions.

b) Correct at appropriate level.

Reg Dell - 10 - comp - 1 - 100 - 1000.  
90 - 900.

PO

Dell - 10 - comp - 1 - 90 - 900.

vchr

Dell - 10 - comp - 1 - 90 - 900.

5) Post vouchers:

This process creates the data that is req. to  
distribute from AP to GL & populates those  
entries into "voucher accounting line table." (2)

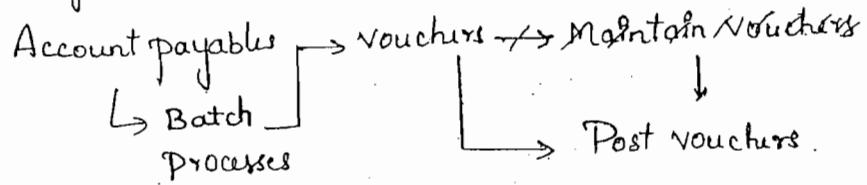
Table that gets populated is

(1) VCHR ~~xxxx~~ - ACC TG - LINE

Process Type: Application Engine program

Process Name: AP\_PSTVCHR

### Navigation:



→ During this process, Sys. creates payment lines depending upon Voucher Type.

a) If it is a withholding voucher, Sys. creates one payment line for vendor & other payment line for Attorney vendor.

b) If voucher consists of any prepaid voucher, Sys. creates one payment line for vendor & other advance payment lines.

### 6) Perform Payments:

This is a place where we can do payments to the posted vouchers. Record: Payment Voucher cross ref. table.

#### Types:

PYMNT-VCHR-XREF

PAYMENT

a) Express payment / Inhouse cheque printing:

This method is used to pay single voucher at a time.

### Navigation:

Account Payables → Vouchers → Add / update  
Create voucher.

### b) Pay cycle:

This process is used to pay multiple vouchers at a time.

One of complex process in Acc. payable.

#### Navigations:

Acc. payable → payments → Paycycle → Paycycle Manager

#### Paycycle steps:

##### ① Execute payment selection process:

During this process, sys. selects all the list of vouchers that are posted & pending for payment according to the criteria specified.

##### ② Select or deselect the vouchers that need to be paid:

This is the place where we can select or deselect the vouchers that are to be paid & selected as per step ①.

This is a manual process.

##### ③ Payment creation process:

During this process sys generates the payment ref. id for all the selected list of vouchers.

##### ④ Print cheques:

During this process sys prints checks for all the

cheque

Date : Payment date		
Payee	Vendor name	
Amount		
RS	[ ]	
"cheque" no	inter code	Vijay

payments for which payment ref. id is generated  
as per step ③.

elements of pay cycle:

a) Step:

Each & Every step in a pay cycle is a process or  
a program. . specifies program type & prgm Name

b) Step group / Step Table:

collection of steps that are executed in same  
sequence

④ Default Step group provided by PeopleSoft  
sys. is named as 'Model'.

c) Paycycle Definition:

This is a place where we specify selection  
criteria like date range, payment Type,  
step table, vendor Type etc.

### Navigation:

For step:

Account payables → payments → paycycle definition  
↓  
Step

For step group:

Account payables → payments → paycycle definition  
↓  
Step group

For paycycle definition.

Account payables → payments → paycycle definition  
↓  
Processing

Payment selection:

Paycycle selection criteria.

Program <sup>Type</sup> Address : App' engine prgm.

Program Name : AP-APY20215 (I)

Payment creation :

Prgm Type : APP' engine prgm.

Prgm Name : AP-APY2015

Print cheques :

Prgm Type : crystal check.

Prgm Name : check3 (or) APY2021

### 7) Post payments :

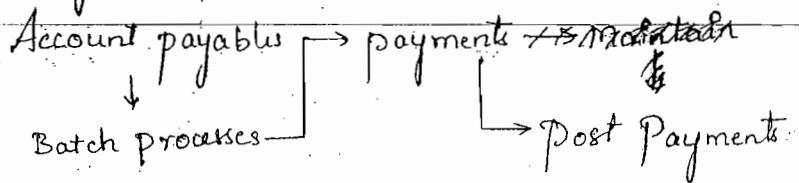
payment posting process : This process selects the data related to payments that need to be distributed to GL and populates those lines into Voucher accounting line table.

process type : App Engine Type

process Name : AP-PSTPYMNT Record :

Navigation :

Voucher Acc. Line table



### 8) Close Voucher :

This is used to mark vouchers as complete after entire processing for that voucher is done & that info. has been distributed to GL.

### Asset Management :

→ Asset management is process driven.

#### Asset :

Item which consists of depreciation/Appreciation

3 years ago - Land in Hyd - 1,00,000

Now

- 2,00,000

1,00,000 } Depreciation  
50,000 }

Business

C  
Y  
e  
a  
t  
e  
A  
S  
S  
E  
T  
B  
A  
S  
E

D  
V  
O  
U  
C  
H  
E  
R

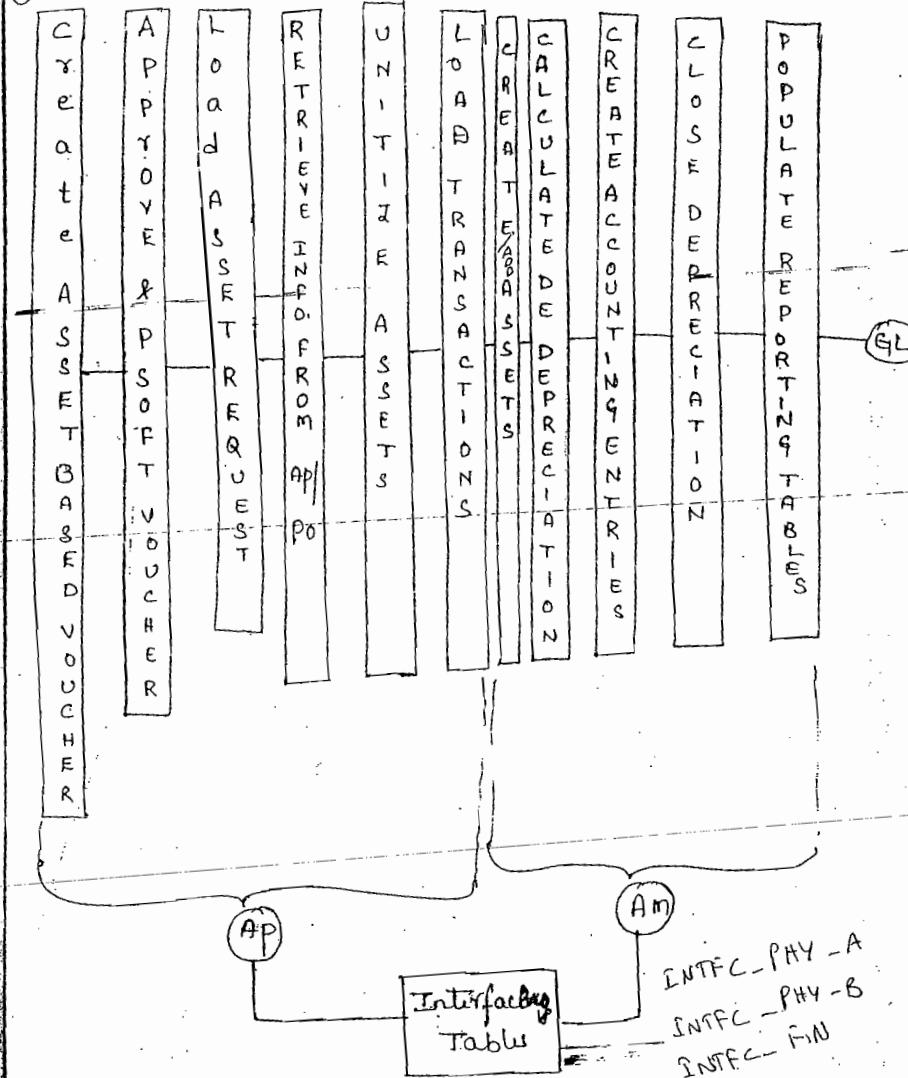
Trans.  
④⑤

Tr  
④⑤

Acc  
④⑤

## Business Process:

Select the  
processes into



## Transaction Date:

Date on which TX occurs.

Transaction: creating a po

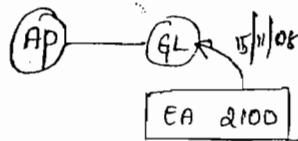
Requisition etc.

## Accounting Date:

Date on which the TX's must hit GL accounts.

### Vouchers

1	01/11/08	EA	100
2	02/11/08	EA	200
3	03/11/08	EA	300
4	04/11/08	EA	400
5	05/11/08	EA	900



This is created in AP & sends to GL

Sys. does not matter how many purchased, it looks only how much amt. of money is spent from this account.

The date 15/11/08 is called Accounting date.

### Acquisition Date:

Date on which assets are acquired.

1. comp. 8/11/08 1000 → comp. we bought  
↑ Tx Date, Acq. Date

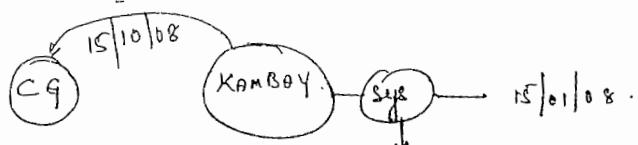
2. comp 8/11/08 01/01/08 1000  
Tx Date Acq. Date. → comp. belongs to another company who assets captured by own

### Inservice Date:

Date on which the asset has started for usage.

	Tx.Dt	Acq.Dt	In.Dt	Acq.Dt
1. comp	8/11/08	08/11/08	15/11/08	8/11/08

↑ usage.



2. comp 8/11/08 08/11/08 15/01/08 15/10/08

### Retirement Date:

Date on which the value of asset becomes zero.

Retirement Date comes from Inservice date.

In the above case the retirement date is 14/01/08

### Asset class:

Collection of Assets which consists of same life or same depreciation %.

### Am Business proc: (Explanation)

#### Create Assets:

Add

#### Methods:

##### a) Online Assets:

These are the assets which are directly added on Am module.

##### b) Assets from AP:

These are the assets which are created acc. payables data.

### Details:

1. Asset ID
2. Asset class
3. TX Date
4. Accounting Date
5. Inservice Date
6. price
7. BOOK

↓  
100 Assets

### Tables:

Asset	BOOK
	Cost
	open-trans (open tx table)

This table specifies whether acc. interest, depreciation, reporting table has been populated or calculated or not.

20 - Tax based Assets → These details are maintained in  
80 - Non " " " separate books.

### Navigation:

Asset Management → Asset Tx's → owned Assets

↓

Comp	life	Inservice	Basic add.
1000	10months	01/01/08	

### Calculate Depreciation:

This process creates entries in depreciation tables which consists of depreciation amount for an asset from inservice date till the life of the asset on monthly basis.

### Methods of Depreciation:

- 1) St. line Depreciation
- 2) Declining Balance Depreciation

straight line Depreciation :

Uses Life

Declining Balance Depreciation.

Uses depreciation %.

		depreciation	Price	Declining
AS1	Jan	- 100	100	100
AS1	Feb	- 100	900	- 90 - 900
AS1	Mar	- 100	800	- 81 - 810
AS1	Apr	- 100	700	- 729
	May	- 100	600	st. line
	June	- 100	500	
	July	- 100	400	
	Aug.	- 100	300	
	Sep	- 100	200	
	Oct	- 100	100	
				0

After Oct we can sell with 0 value.

Navigation:

Asset mgmt → Depreciation → processing

↓

Calculate

DEPRECIATION is the table that is populated.

Program Type :

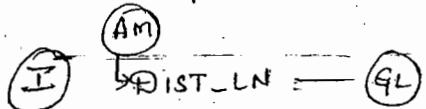
8.9 — App' Engine ; < 8.8 — SQR

Program Name :

8.9 - AM\_DEPR\_CALC, < 8.8 - AMDP CALC

## Create Accounting Entries:

This process creates entries into asset distribution line table which consists of data that need to be send to GL related to asset addition & asset retirement tx's.



### Navigation:

Asset Management → Accounting Entries

↓  
Create Accounting Entries.

### Process Type:

8.9 & 9.0 → App' Engine, L=8.8 → SQR process

### Process Name:

8.9 & 9.0 → AM\_AMAEDIST, L=8.8 → AMAEDIST

## Close Depreciation:

This process creates accounting entries for the asset depreciation tx, & populates the data into Asset distribution line table for the current month.

### Navigation:

Asset mgmt → Accounting entries → Close Depreciation.

## Process Type:

8.9 & 9.0 — App' Engine

<= 8.8 — SQR process

## Process Name :

8.9 & 9.0 — AM\_DPCLOSE

<= 8.8 — AM\_DPCLOS

## Populate Reporting Tables:

This process populates the data which is required for reporting into Asset mgmt for reporting tables.

### Navigation:

Asset mgmt → Financial Tables Reports

↓  
Load reporting table.

↓  
Dep. reporting table.

Process Name : AMDPREPT

Process Type : App' Engine

Populates a table called DEPR\_RPT\_TBL.

## Create Asset Based Voucher:

This is a voucher created against asset items. By creating these vouchers we need to select a check box called as asset on the Voucher creation page.

### Navigation:

Account payable → vouchers → Add/update

↓  
Regular Entry

### Load Asset Request:

asset based  
This process selects the list of vouchers that are not yet sent to AM.

### Navigation:

Account payable → Batch processes → extracts & load

↓  
load Asset Request

### Process Type:

App Engine prgm

### Process Name:

INTFAPAM

### Retrieve Info. from Ao/Po:

This process is used to select the data from the acc payable tables that are req. for asset mgmt module & populates this info. into asset mgmt interfacing tables.

### Records:

INTFC-FIN

(1) INTFC-PHY-A, INTFC-PHY-B.

INTFC-FIN — Financial info. table.

↳ consists of cost of assets.

INTFC-PHY-A } — physical info. tables.

INTFC-PHY-B }  
Eg: Serial no., etc.

10 comp = 10.00

↑  
Group Assets.

We should not have group assets we have to break it into individual assets.

Unitize Assets:

The process of breaking a asset where the quantity  $> 1$  into equal individual pieces is called as Asset Unitization.

Composite Asset / Group Asset:

An asset where the qty  $> 1$  is called

Group Asset

Load That's back

Navigation:

Asset mgmt  $\rightarrow$  Spend / Receive info  $\rightarrow$  unitize assets.

## Load Transactions :

The process of creating assets in asset mgmt with the data from the asset interfacing table is called as Tx loader process.

## Navigation :

Asset mgmt → Send/Receive info → Load Tx's

↓  
Load Tx's into AM.



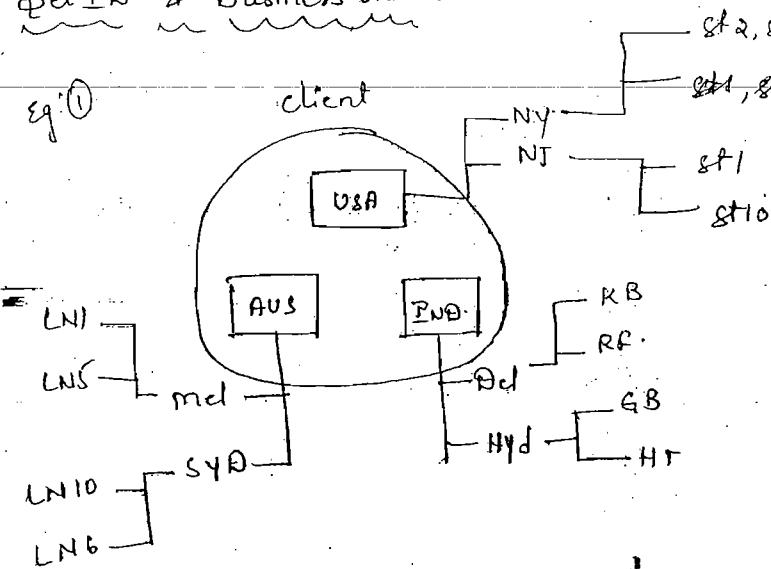
process Type : App' Engine program

process Name : AMIF1000



## General Ledger :

### Set ID & Business unit :



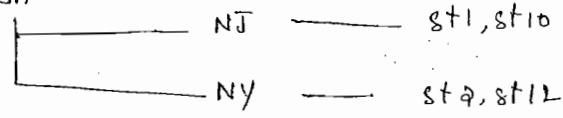
Act as key field for control Task

defines

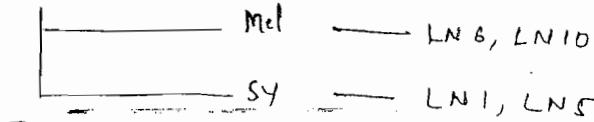
structure of client business.

Set I Business unit location/site

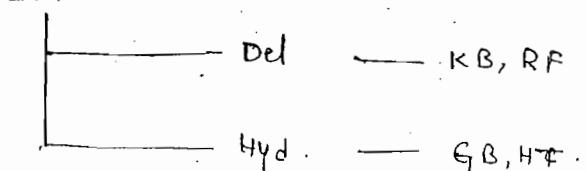
USA



AUS

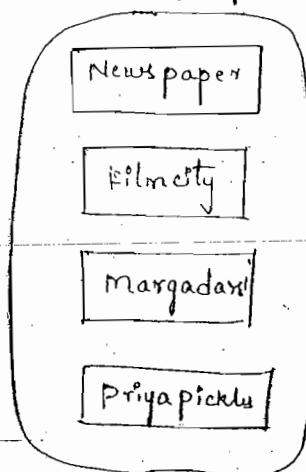


IND



Eg(2):

Eanadu groups



Set II

BU

sites

Eanadu Group

NP — All

FC — Hyd

MD — All

PP — Gurur

## Calendars & periods

Setup Fin: supply chain

↓  
common def → calendars / schedules

↓  
calender builder

Eq:

IND

Jan - Dec.

April - March 31 - Financial calendar.

2008

IND → April 01, 2008 To March 31, 2009

USA - Jan 01, 2008 To Dec 31, 2008.

→ In India yearly once we pay the Taxes

→ calendars are divided into periods -

Each day is period, week, month.

2008 → 365 — daily calendar - 365 periods

- 52 weeks — weekly calendar - 52 periods

- 12 months

- 4

→ 2008

①  
Financial  
Calendars  
periods

	IND	Period	USA
1	Apr, 08	1	Jan, 08
2	May, 08	2	Feb, 08
3	June, 08	3	Mar, 08
4	July, 08	4	Apr, 08
5	Aug, 08	5	May, 08
6	Sep, 08	6	Jun, 08
7	Oct, 08	7	July, 08
8	Nov, 08	8	Aug, 08
9	Dec, 08	9	Sep, 08
10	Jan, 09	10	Oct, 08
11	Feb, 09	11	Nov, 08
12	Mar, 09	12	Dec, 08

997, 998, 999 - Adjustment periods.

92

orders/schedule  
↓  
Under Builder

### Chart Fields:

These are nothing but the normal fields that are exists in the people soft systems.

These fields are used to maintain the security in people soft sys.

We have 21 chart fields in financial system.

18 - Delivered chart fields.

3 - custom chart fields.

### Chart Field List:

- 1) Account.
- 2) Alternate Account.
- 3) Operating unique.
- 4) Fund code.
- 5) Department.
- 6) Program code.
- 7) Class Field.
- 8) Budget reference.
- 9) Product.
- 10) PC Business Unit  
    ↳ project costing
- 11) Project.
- 12) Activity.
- 13) Source Type.
- 14) Category.
- 15) Sub category.
- 16) Affiliate.
- 17) Fund Affiliate.
- 18) Operating unit Affiliate.

### Nav:

Setup Financials/Supply chain



Common Definitions



Design chart fields



Configure



Std. configuration

## Chart of Account

This is a table which is created using chart fields.

This table is created by running a process called as chart field configuration process.

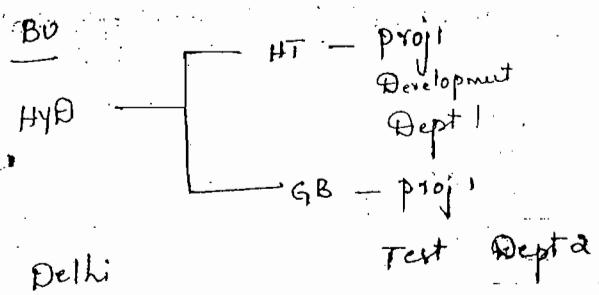
- This table stores valid chart field combination data.

### Advantage:

Whenever any entry is being created in the people soft financial sys. (like PO, requisition, voucher, asset) sys verifies with the data available in the chart of accounts.

If the chart field data specified in the entry, does not match with the data available in the chart of accounts table then sys will not allow us to save the entry throwing a msg

### 'Invalid chart field combination'



project mgr. raises req. for 10 sys.

### Chart of Accounts

proj	Dept ID	operating unit
proj	Dept 1	HT
proj	Dept 2	GB

{ Dept ID - Dept 2 }  
proj - proj } This will not allowed by  
OU - HT chart fields.

checks data with chart fields data.

Mandatory chart fields : chart field configuration.

a) Account      Process Type: App Engine

b) Department      "      Name: FS\_CF\_CONFIG

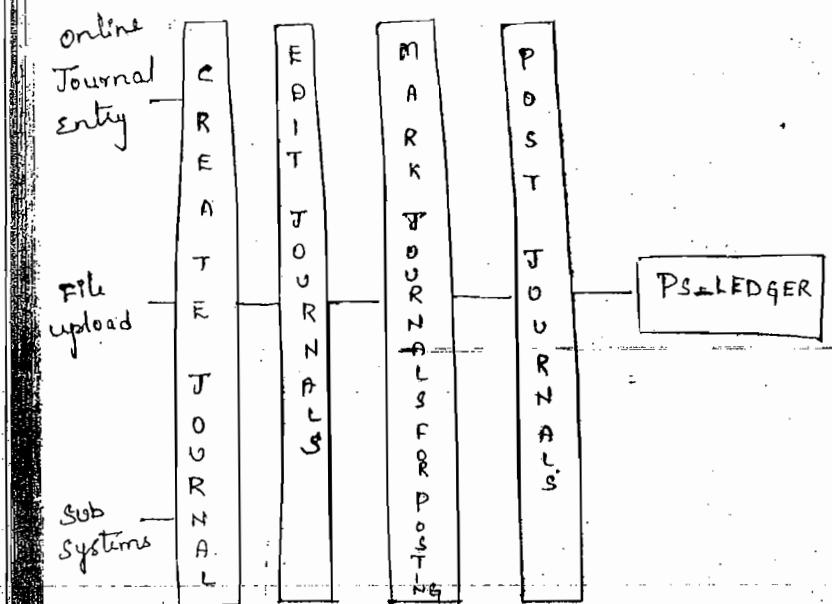
c) Project ID or product ID      Place where we define  
    chart field comb.

Business Process of GL:      setup Fin./Supply chain →  
    common def → design chartfield

Journal :      comb. Rule ← comb. editing

The entry what we created in GL is

called Journal.



### Create Journals:

#### 1) online Journal entry :

Types:

1. Standard Journals

2. Reversal "

3. Adjustment "

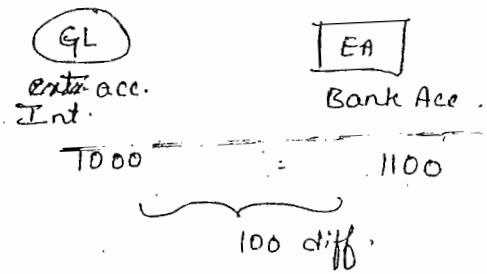
4. Recurring "

5. Suspense "

→ Normal Journals is called standard Journals.

→ Tallying internal accounts (GL) with ext account (Bank A/c) is called Reconciliation.

- Adjustment Journals are used to reconcile internal accounts with ext. accounts.
- we know Account, Financial year / physical year, period etc.



- Reversal Journals are used to reverse amount in Journal.
- Recurring Journals occur frequently for a given time period.
- Suspense Journals are created, to tally or to balance Internal accounts with ext. accounts. These amounts will be charged to miscellaneous accounts and to adjustment periods.

Navigations:

General ledger → Journals → Journal entry.

↓  
Create & update Journal Entries.

### a) File Upload:

Import  
a) Flat File upload : GL → Journals → Import Journals

↓  
Ext. flat file

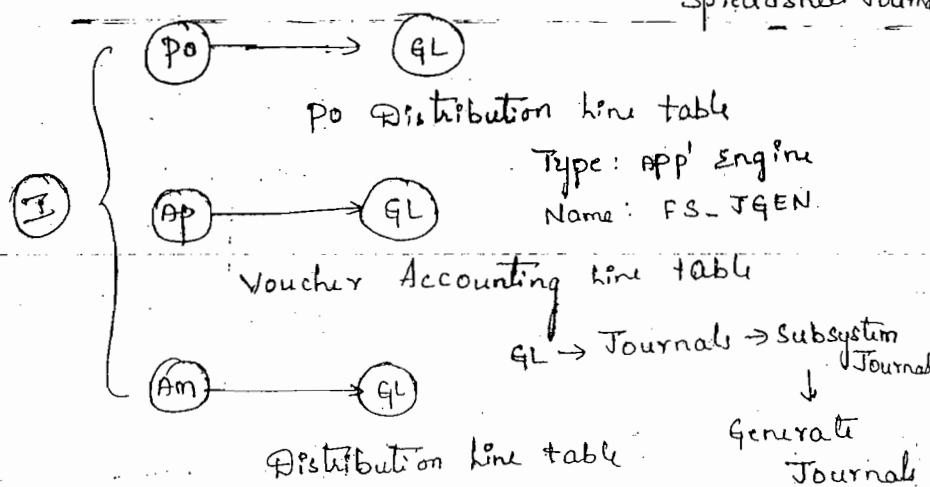
This is importing Journal data from a flat file (or) a text file into the peoplesoft sys.

## b) Spread Sheet Journal Import (SSI)

This is importing Journal data from a XL file

(or) a spread sheet into people soft sys.

↓  
3) Sub system Journals : GL → Journals → Import Journals  
↓  
Spreadsheet Journals



The process of getting Journal data from sub systems like AP, AM, PO, AR etc is called as Sub system Journals.

The process used to create Sub system Journals is called as Journal Generator.

Edit Journals:

The process of marking Journal as complete after performing the req. data changes.

- a) Single Journal Edit
- b) Multiple Journal Edit

### a) Single Journal Edit :

we can only edit one Journal at a time

#### Navigation :

GL → Journals → Regular Entry → Add/Update

### b) Multiple Journal Edit :

The process of editing multiple Journals at a time.

Process Type : App Engine

#### Navigation :

Name : GL-JEDIT

GL → Journals → process Journals

↳ Edit Journals .

### Mark Journals For Posting :

→ This is a manual process where we can select and mark multiple Journals that need to be posted.

#### Navigation :

Journals → process Journals

↳ mark Journals for posting

### Post Journals :

JRNL - HDR → Journal Header Table

JRNL - LN → " Line " } upto before  
} posting the

Journal the data is stored in these tables.

→ The process of sending Journal data from Journal header & line tables to ledger table is called as posting Journals.

### a) Single Journal Posting:

we can only post one Journal at a time.

Navigation:

GL → Journals → Journal entry

↳ create / update Journal Entries.

### b) Multiple Journal Posting:

we can post multiple Journals at a time

Navigation:

GL → Journals → process Journals

↳ post Journals

→ The app's security in financial sys. is maintained using chart fields as well as user preferences.

User Preferences:

process Name : GLPPPOST

" Type : COBOL SQL process

Navigation:

Set up Financials | Supply chain → common definitions

↓

User preferences

↓

Define user preferences.

User ID : begins with  

### Overall preference:

This is a place where default values are specified whenever an entry is created in the financial system.

click on overall preference.

### Procurement:

click on this

User : VP1

Loc :

Origin :

Department :

Ship to Loc :

Requestor :

Buyer :

These are all default values.

In this we have purchase order Authorization.

Approve

cancel

If we dispatched the PO but we don't want those, then we can cancel.

### Delete

Before dispatching or approving we can delete it.

### close :

→ Cancel :

→ This is done after po dispatch process.

→ The data will be available in po table.

→ Delete :

→ This is performed before po dispatch (or) po approval process

→ This will remove data from Po tables.

→ close :

→ This is performed after entire operation  
on that po is complete.

→ Data will be available in po tables.

can work Approved Po's

Full Authority for All Buyers

### Requisition Authorization :

Approval

cancel

Delete

close

### Payables online Vouchering:

Pay unmatched vouchers Pay unmatched :  (1000)

If this is selected, even if there is any unmatched occurs in price or amount it allows.

1000 means upto min 1000 diff it allows.

### Receiver Setup:

Interface Receipt

get data from po & create receipt.

### Vendor processing Authority:

Approve Authority to enter

" " Approve

" " Inactive.

→ people Tools → process scheduler



process monitor

we can see all the process status.

### Reports in purchase order:

#### i) Print Requisition:

Used to print a requisition

#### Nav:

Purchasing → Requisition → print requisition.

Process Type : SGR Report

" Name : PORQ010

## 2) Po Requisition Xref :

This is used to print requisitions & their related po's.

Nav :

Purchasing → Requisition → Reports

↓  
Po/Requisition Xref

Process Type : crystal Report

" Name : POY1100

## 3) Po status listing :

This report gives list of reports & their current status

Nav :

Purchasing → purchase order → Reports

2 Process Types :

Status Listings

a) listing By vendor

Type : crystal, Name POY4020

b) listing By item.

" " crystal, " POY4021

## 4) Receipt Delivery Report :

Used to print receipts.

Proc

NAV

PUR

Repo

Vouch

char

Proc

NAV

AP

Match

Thi

match

Proc

( )

NAV

AP

Posted

The

Process

Process Type : crystal  
Name : Poy5030

Nav:

purchasing → Receipts → Reports → Delivery Reports

Reports in Account payable:

Voucher Listing By chart field:

This gives the list of reports which has chart field combination.

Process Type : crystal

Name : APS8003

Nav:

AP → Reports → Vouchers → Voucher Listing By chart field

Match Exceptions:

This provides a list of vouchers consisting of match exceptions.

Process Type : crystal

Name : APY1090

Nav:

AP → Reports → Vouchers → Match Exceptions

Posted Vouchers:

This gives a list of posted voucher

Process Type : crystal process Name : APY1020

Nav:

AP → Reports → vouchers → posted vouchers

Vendor Detail Report:

This gives the details of vendors.

Process Type: crystal

Name: Apy3000

Nav:

AP → Reports → vendor → Vendor details

Open Prepayment Report:

This gives the list of advance payments which are still in open status.

Process Type: crystal

Name: Apy2100

Nav:

AP → Reports → prepayments → open prepayments

## HRMS

### PeopleSoft Applications:

→ PeopleSoft Apps are developed by people tools

→ Some of the ps. App's are

1) HRMS (Human Capital Management System)

2) FIN & SCM

[Financials, Supply chain, Enterprise Service Automation]

These 3 are part of FIN & SCM

3) Campus Solutions

(Student Administration)

Now-a-days, this comes under HRMS

4) ELM (Enterprise Learning Management)

5) CRM (Customer Relationship " )

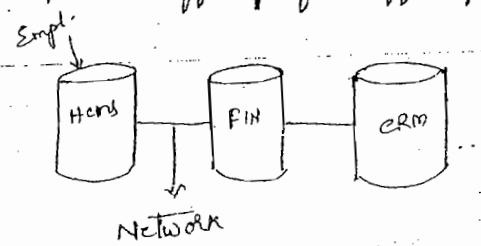
6) EPM (Enterprise Performance " )

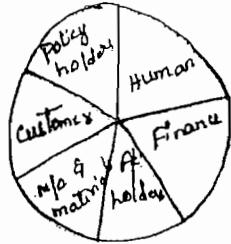
7) Enterprise portals

→ Each of these app's have diff. db.

→ Oracle - Single db for all app's

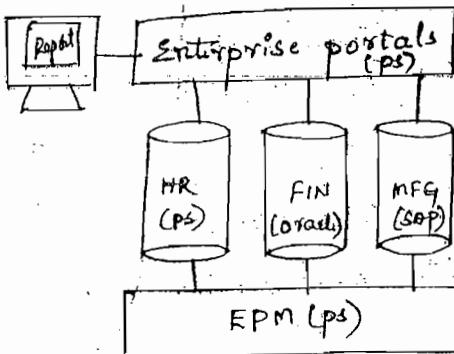
ps App's - diff db for diff app's





→ we have diff. app's for diff. ERP's.

→ ERP package have delivered integrations to integrate diff. app's with ERP's just by configuring



Epm contains dataware housing tool

→ Using enterprise portals we are not able to get reports from db's but it is possible with EPM

→ Using enterprise portals we can create our own portals.

→ ps offered a complete business solution to the enterprise.

→ The highlevel structure is same for all app's.

→ APF

PS A

Hcr

→ con

the U

the T

cont

→

+ t

→

①②

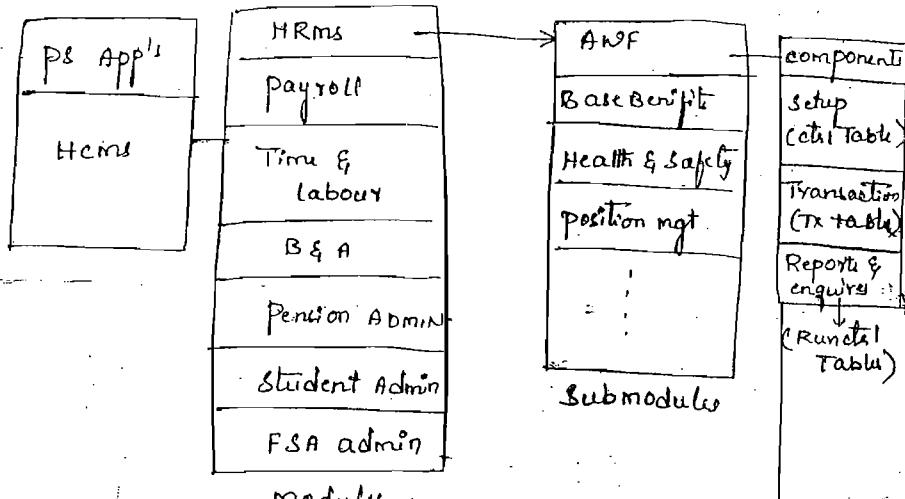
③

④⑤

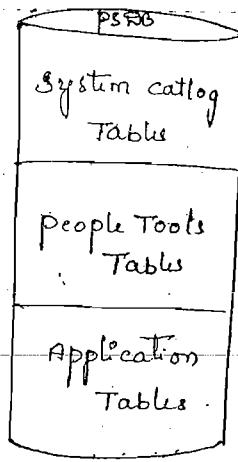
⑥

⑦⑧

→ App' consists of so many modules.



→ control tables will give the list of valid data for the Tx tables.



### Control Tables :-

→ The above 3 tables (ctrl Tables, Tx tables, Runch tables) are all connected to App' table.

→ The data we can maintain in control data is

#### ① Master Tables data:-

Master Tables contains static data (or) semi static.

#### ② Control data will maintain Accounting &

Organisational structure.

#### ③ Rules, Regulations, policies & procedures.

## Transaction Tables

Type of data stored is

- Day-to-Day Business Transactions
  - Dynamic data.
  - Tx tables are very important.
  - majority of SQR reports source is from Tx tables
  - " " crystal " " " " et al "

ANF - Administration with Force.

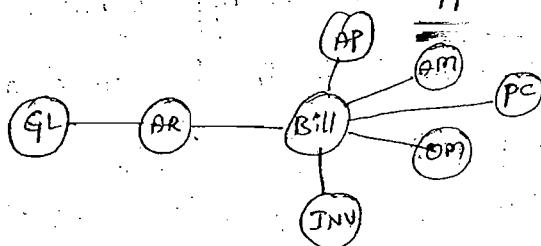
Any ERP work is divided into 3 types

- ① Implementation:
  - ② upgrade. ( Versions  )
  - ③ Support, / Enhancements: \ patches & fixes

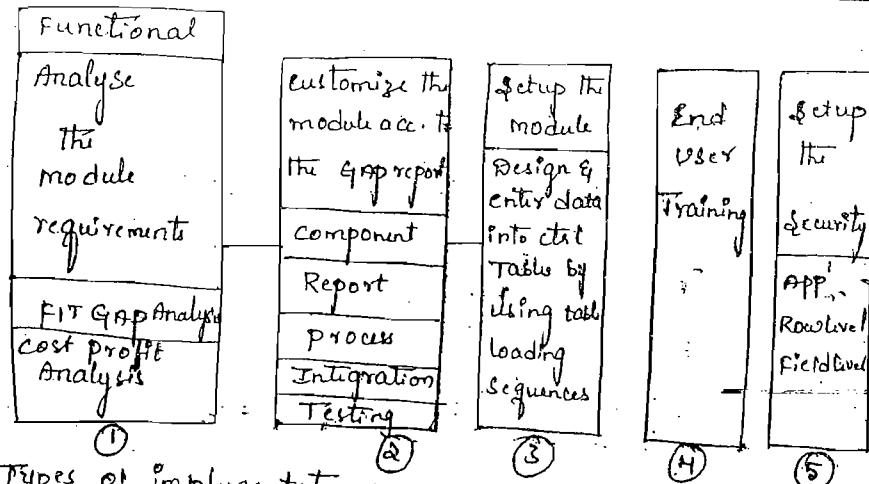
## Implementation lifecycle : (module wise)

- Each & every module is independent in ps App's

Each module is a small app!



- If all the modules are not implemented also, what if the info. we are getting from diff. modts we have to enter it on-line.



### Types of implementation:

#### 1) Vanilla Implementation

Following the best business practices by the ERP.

Changing the business process without changing ERP.

Customization is in report level.

#### 2) Customized implementation : Changing the business process of ERP as per client. Cost profit Analysis :

Filling gap cost & the upgrading cost we have to find.

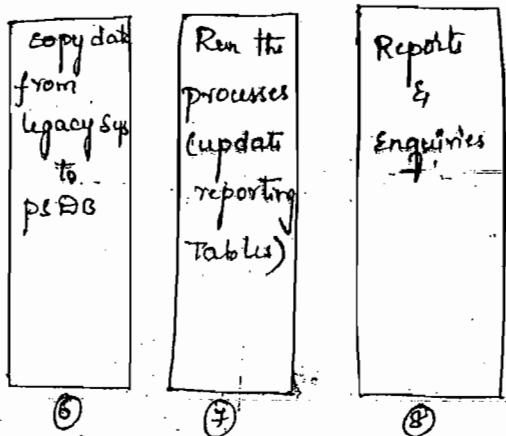
Filling gap cost → depend upon the technology cost differs.

cost depends on the benefits.

Upgrade cost depends on the customization.

→ Fitgap Analyser's need to analyse the business process of client with the business process of ERP.

- Design data from high level to low level
- we can't generate the table loading sequence.
- Field level security is obtained by people code.



Reporting Tables — we get data by running some process, we cannot enter data online.

→ Success of implementation depends on ① step.

### Role

- ① Analysis Functional
  - ② customization Technical
  - ③ Setup the module Functional
  - ④ End user Training Functional
  - ⑤ Setup the security Functional / Technical
  - ⑥ copy data " "
- Data mapping      develop process

⑦ R

⑧ F

→ In

Termin

Term

Glob

① Fi

② Du

③ B

④ Se

⑤ Eff

Modu

① C

② K

③ C

④ V

⑤ P

⑥ O

⑦ L

⑧ S

⑦ Run the process      Functional

⑧ Reports      Functional

→ In whole project there is 7:3 ratio

### Terminology:

Terminology is divided into 2 types.

① Global

② module specific.

#### Global :-

① Table sharing

② Data sharing

③ Business unit

④ Set ID / Record group / Table set control.

⑤ Effective date / eff status / eff sequence.

#### Module Specific :- (HCMs)

① Company

② Regulatory Region -

③ Establishment

④ Location

⑤ Department

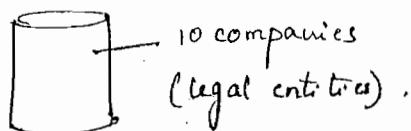
⑥ Job code

⑦ pay group

⑧ Earning & Deduction codes.

Table Sharing: → PS supports table sharing in all tables

To identify control tables - Set ID  
" " Tx " - BU. { key fields

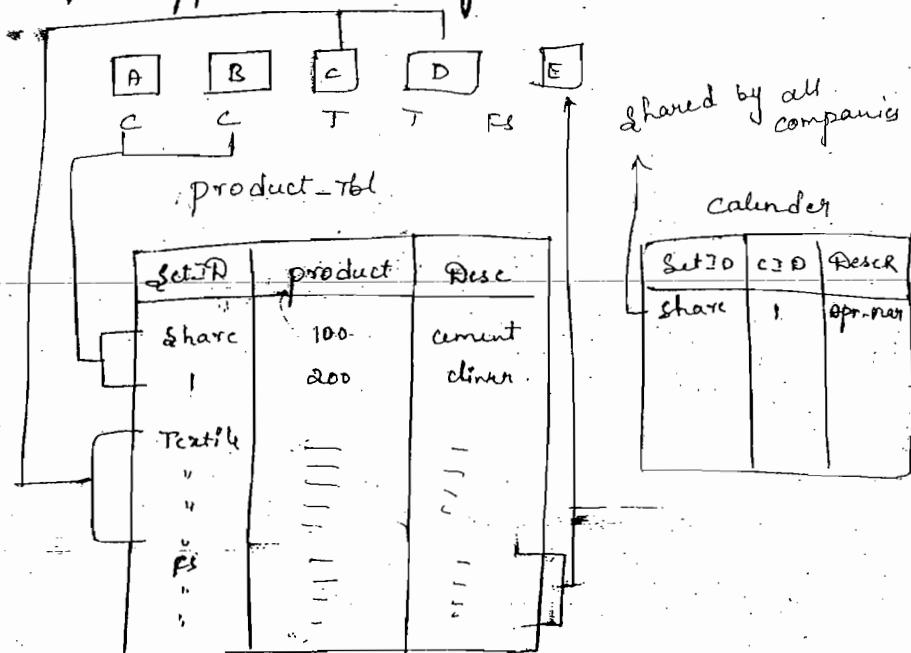


Dept-Tbl

we maintain only one dept-Tbl for all the 10 companies

Data Sharing:

→ PS supports Data sharing in Control Tables



→ Data sharing is not possible in all cases.

→ " " is used to reduce redundancy.

→ Eg: Rules, Reg, policies & procedures can be shared among all companies.

In all tables

→ we can find control tables without SetID

e.g.: country-Tbl, stat-Tbl, currency-Tbl.

→ where the data is unique for all companies then

SetID is not req.

where the data is not universal we have to use SetID.

Eg: Countries will not change from Company to company, so SetID is not required.

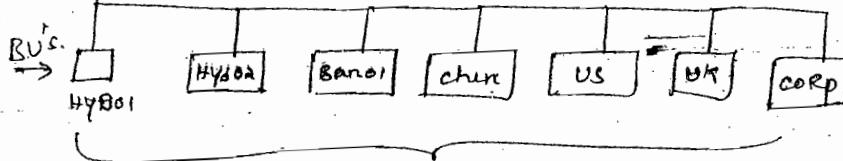
→ Data sharing is never possible in TX tables.

Business Unit: → length is 5.

→ High level key field of TX table.

→ BU represents set of data of a legal entity  
" " " " " logical entity

↳ REPORTING company → legally identified company.



logical Business units

Pivot Sys INC

90 sub (US)

} BU

Pivot Sys IND LTD (company in India)

Bang  
chu  
Nodis

} operating units

→ Based on the report requirement, we can define the structure of BU.

Nav:

Finance:

To create a BU.

Setup fin & supply chain → Business unit related

General ledger

General ledger Obj

HR:

Setup HRMS → Foundation Tables → organisation

BU

① In HR, to identify legal entity, company is used.

② In FIN, for each & every module we have to define separate BU's.

But in HR, Once we create a BU, it is available

for all modules.

Set ID:

→ High level key fields of ~~Set ID~~ control tables.

→ useful for to identify the ~~Set ID~~ rows in a control Table as 'Sets'.

→ Set ID length is 5.

\* Req:

→ ~~Set ID~~

can define

### Dept-Tbl

SetID	Did.	Reser.
Share	—	—
Textile	—	—
PS	—	—

→ Each set represents to set control value.

majority cases " " " is BU.

To Define SetID.

### Finance:

① while creating a BU a SetID is with the name of BU will be created automatically.

② peopleTools → utilities → Administration



Table SetID

### HR:

peopleTools → utilities → Administration



Table SetID

### Record Group:

→ Represents a group of control tables having similar functionality.

Nav: pT → utilities → Administration → Record group

## Table Set Control

- set control values represents sets of control data across the db. (or)
- represents a set of data on each control table across the db.

Nav:

PT → utilities → Administration → Table set control

Eg:

ABC-Rec:

BU	calc Q	Acc Q	Dept Q	product Q	Amount
0001	01	400000			

St-Account set

Dept-Tbl

Dept-Tbl

fact ID	Dept ID	Date
share	=	=
Txt	=	=

we can see only the set of values belongs to one set ID.

Go to Record properties

↳ Set control field

Business-unit

0001

Record group

Fs-05 Department share

→ If ABC-Rec does not contain BU then

Record field properties → set control field.

Give any field name of ABC-Rec

with

with

SetID - AUS01.

Record group - FS-05. (Accounts)

In HRMS we can  
find 40 record  
groups.

FS-06 (Departments)

Table set control.

Set control value : US001

Journals → Journal entry

↳ Create / update Journal entries.

→ If NO SetID for a table we will not find in  
Record groups.

11/10/08

Effective Date / Status / Sequence:

↓ Main purpose of E.D is to maintain the data  
in a table as history, current & future.

Emp-Rec

1/1/04 1/1/06(salt) 1/1/07 1/1/09

SetID	K	share	share	share	share
Emplid	K	2500	2500	2500	2500
Eff-Dte	K	1/1/04	1/1/06	1/1/07	1/1/09
status		A	A	A	A
B0J		1/1/04	1/1/06	1/1/07	1/1/09
Banc		5000	7000	2000	10000
TA		4000	5000	5000	7000
PA		8000	8000	8000	6000
HRA		2000	2000	2000	3000
ContactNo:		65576420	65576420	65576340	65576340

Without Eff-date field it will override the info.  
with the new data.

→ we can maintain 'N' no. of history & future rows  
but " " " only one current row.

### Conditions to find the current row

- ① Nearest to sysdate & less than or equal to system date

Effective status :

↳ Have 2 values

① Active (A)

② Inactive (I)

Emp-Rec  
1/6/09

BU	ASSETID	ASSETFEE	EMPLID
UL001	10506		

Share

2500 → This person is going to

1/6/2009 resign on 1/6/09. so we should not set him

1/6/09

10000

7000

4000

8000

66495320

Suppose if we are using Emp-Rec as

prompt table. In drop down list his

Emplid should not available in the list.

For that we have to set the status

(status) I

Effective Sequence :

Eff-date is a Date Field.

→ we should not add 2 rows with same Eff-date

Seq - K

1st action → Seq '0'

Eq:

If

balance

all

we

Eq:

If

balance

all

we

Emp-Rec

bco

field

That

If

it

C

C

U

O

future rows

2<sup>nd</sup> action — Seg 'i'.

Eq:

If a person is transferred, promoted & his salary also incremented, then we need to enter all the 3 details in 3 rows. For such cases we use eff-seg.

Eq:

sysdate - 5/1/2009

Dept-Tbl

DeptID	K	Share
DeptID	K	10000
Eff DT	K	5/1/2009
Eff status	A	MR
Descr		MR
Loc		Hyd
Mgrid		

Emp-rec

→ Actually the row displayed should be  
5/1/09.

But the row displayed here is 1/1/07.

b'coz for this Dept-Tbl also it has eff-dt field. In that case mgrid prompt table takes that Eff DT row as system date.

If Dept-Tbl does not have eff date row, then it will take actual system date.

## Module Specific (Hcms)

Company: will be a pay entity.

This represents legal Entity.

BU represents logical Entity.

Legal Entity may be a ltd ; pvt.ltd , proprietorship, partnership.

Regulatory Region:

This represents a set of rules & regulations.

Belongs to

- ① profit Laws / Benefit Law
- ② Labour Laws

Establishment:

IS mandatory in USA.

EEO - Equal Employees opportunity act.

Each & Every office is establishment.

Location:

Location is having link with regulatory region;

holiday calendar, Salary plan

Department:

In Hcms, Department is used mainly for row level

Security .

JA  
JOB C

Eq:

(M)

PS

Pay P

BP

CR

DP

Q

RR

HR

Earnit

Q

Q

Q

Q

Q

Q

Q

XA  
JOB code :

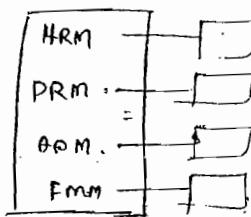
e.g: Training module.

Recruit "

A

Manager

B



PS supports position as well as JOB codes  
govt orgn put

Pay Group:

↳ group of employees

criteria

- ① pay frequency → main criteria
- ② country
- ③ currency
- ④ Regulatory Region

Earning & Deduction codes:

## Administrator work force

### ① Setup the module (AWF) :

① Global setup — ctrl Tables

② Setup specific to Administrator wf. — ctrl Tables

### ② Understand the Transaction Business process.

↓  
Tx tables / include & things

Run control table  
① Transaction  
⑥ Process  
Validation  
Update  
Reporting

③ Reports & Enquiries  
Run control  
Tables → Tx tables.

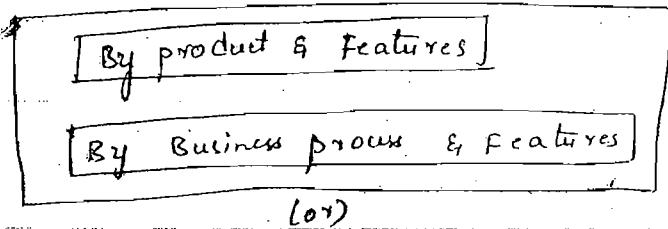
## Setup the AWF

Generating the table loading sequence.

Enterprise components → manage implementations

↓  
Manage config set

Create config set



Retrieve config set

Config set [AWF]

click on By product & Features.

HR → view summary.

Delete the unnecessary wf manually by clicking on .

Save

Config set [HR-AWF]

Name

[HR-AWF]

OK

Generate setup Tasks

One process will execute.

Run status	Ran	Refresh
config set	HR-AWF	
Run ctrl id	HR-AWF	
process instance	519	

View setup tasks.

→ This will get enabled.

↓ after processing is complete

click on Download icon, it generates o/p in Excel sheet.

### ① Installation Table

#### Payroll

- ↳) Global - payroll engine is empty, — Any country
- ② North America payroll - payroll engine is filled with rules & regulations.

Payroll interface gives the reg. i/p's which are useful to run the payroll by 3<sup>rd</sup> parties.

Length of Employee ID field is 11.

#### Third party systems Tab

##### Sys. Defaults

Max. No. of rows in formats 25

Commit after Empl processed: 300

↳ mainly for AE programs.

#### ② Country Table

##### Eg: Addresses Format Tab

Address 1

House No.:

Address 2

Colony Name:

21/10/08

Set up HRMS → Foundation Tables

Currency & Market codes.

— currency Exchange Rate type.

- currency code
- currency quotation method
- currency exchange calculator - Just used as a calculator.

Direct  Indirect

USD → INR

If we choose Indirect no need to define INR → USD.

For Direct we have to define both INR → USD & USD → IND.

#### Market Rate Definition.

Daily we have to update the current rates in this. Otherwise run an automatic process using Integration.

We can set the maximum variance. If we provide this if we wrongly enter the rate then it displays an error msg.

#### Market Rates.

Setup HRMS → Foundation Tables



Organization.

Business Unit.

→ Hydol.

control.

Copy the data of Gbbu to our business unit.

[check whether it is copied (or) not]

people Tools → Administration

↓  
Table set control.]

→ Business unit options Default

Set ID :

Search ↴

Contains the default value to speed up the transactions.

— GL Business Unit

EIP - Enterprise Integration pt

we have to make this as active

Admin people know how to make it active.

If GL data entered in Financial need to be used in HRMS we have to make EIP as active.

— Company Legal Type

Consists of legal entities according to USA

we have to define legal entity according to the rules in India.

Set ID : Rplnd.

we have to map entity to Record group.

[For mapping

People

Ident

C

C

C

In

C

C

In

P

Go

C

E

Go

— Col

C

C

SP

— Lot

C

— Def

O

13  
PeopleTools - Administration - Table set control.

Identifying control Table is in which Recordgroup.

SetID kplind.

ctrl + F.

copy the page Name

In App' designer → open the page.



Find the Table name.

View - Find Ref. References.

In Results we find

Record Group: HR-II.

Go to PT → Admin → Recordgroup:



HR-II.

Go to PT → Admin → Table set control.



Find HR-II → Remove Set ID as

Share

& write kplind.

→ Company

IND

Specify the req. Values.

→ Location

SetID → Kplind.

→ Departments

SetID : share

No managerid & no location means it is a logical dept.

- Org Defaults by permissionlist used for providing security

### [Special permission lists]

- PT → security → userprofiles  
↓  
User profile

[ps]

### \* Permissionlists

- ① Primary
- ② Row Security
- ③ Navigator Homepage
- ④ process profile

In peoplesoft Menu is the Navigation we call on it as base portal menu.

From 8.40 → Base portal Technology

### Enterprise portals

↓  
This is a separate portal

If we have permission we can create our own portal.

- ① portal Menu Navigation
- ② popup Menu

std menu's are not using for Navigation

Purple

③ N.C

④ A.C

Navi

↳ ↴ ↲ ↳

Sou

A.F

PO

Navi

W

In

P.S

on

Acti

Eg

Thru

In

Eg

Add

purpose, they are used for security purpose.

(3) Navigator Home page.

(4) Activity Guide

↳ used for HRMS.

Navigator Home page

↳ is a combination of Business process & Activity source.

Activity Guide — source is Activity.

portal Menu Navigation — portals.

Navigator Home page

worklist → Navigator.

In this navigator home page the sequence of steps is defined.

we can move to next step by just clicking

on Next.

Activity Guide :

Eg Recruiting — create New Job opening.

write any data. [continue]

There we can find hyperlinks.

In HRMS we can have Activity guide navigations.

Eg:

P → Process Schedule → System process Requests

As a user we must have access to change

Server name, Recurrency, we get these permission from process profile permission lists.

In Finance — user preference permission list

In HRMS — primary permission list.

→ Job Attributes.

↳ Job code Table.

— Job Family Table.

↳ while defining Job code we have to specify the Job family to which it belongs.

— Job Function

— Responsibilities

— Job code Task Table

— Job Task

↳ we have to assign the Job Task while defining Job code Task.

— Job profile.

↳ Competencies, Qualifications, Functions etc.

we have to define all these & give a profile ID.

This profile ID's need to be given in Job code Table.

→ Personal

— citizen status

we have diff. citizenship status

permissions

### -National ID Type.

For India we don't have National ID.

- Control Data is used for ass entering valid transactions.
- Compensation Rules.
- Default frequencies by country.

[IND]

- D - Daily
- W - weekly
- M - monthly
- A - Annually

- comp rate code table.

→ common definitions.

- Banking.

### Specific TO module:

Setup HRMS → product Related → AWF.

In AWF (company code)

Actions & Action Reasons are important.

Without AWF there is no other module.

AWF → Add new employees & To maintain the existing employees.

Eg:

Leave is a action

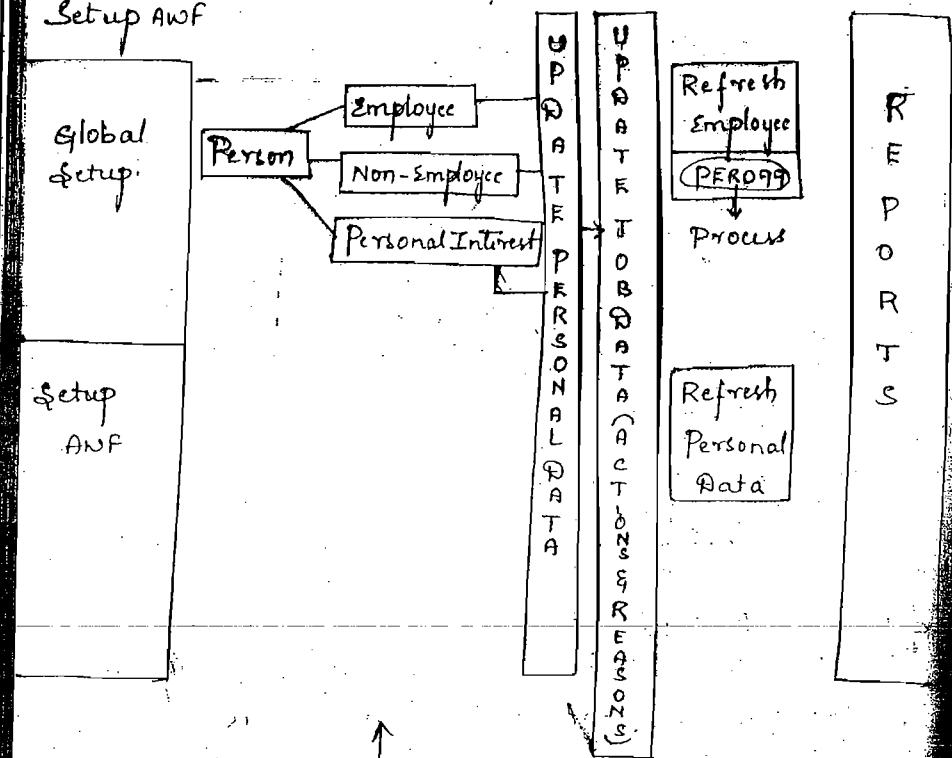
↳ Reason is different.

Sys. maintains some imp. reasons which are used for reporting.

ps defined 78 reasons actions.

### Administer Work Force:

#### Setup AWF



#### Navigations:

① WF administration → personal information

Add a person.

(he can find 'Add a person' in many nav.

Eg: in Biographical)

(Non-Employee → contingent)  
Employee

## ② update personal Data - (components)

WFA → personal Info → Biographical

- ↓
- Bank accounts
- Driver's license Data
- prior work experience
- general comments
- volunteer Activities.

we have to enter  
these informations

WFA → person Info → personal relationships

[To provide Dependent] Dependent Information  
details. ↓  
Dependent Identification Details  
Specify Emergency contact.

WFA → personal Info - citizenship

- ↓
- Identification Data
- passport/visa Expiration
- Audit.

WFA → personal Info - Disability

- ↓
- Disabilities

## ③ update Job Data.

WFA → Job Info → Job Data

X Add additional assignments

### Ⓐ To update Reporting Table

Setup HRMS → Sys. administration → Database  
processes

- Refresh Employees Table
- Refresh personal data

### (5) For Reports

WFA → Job Info → Reports

- personnel Actions History
- Employees on leave or absence
- Temporary Employees
- Years of Service

WFA → Workforce Reports

- Employee Turnover Analysis
- pending future actions
- Request Ad hoc process

(\* under this we can find many

processes from per001 - to person)

Competency Management:

Setup

Global  
Setup

Setup HR

↳ comm  
definition  
→ competencies  
→ accomplish

C

C

C

C

C

C

C

C

C

C

C

C

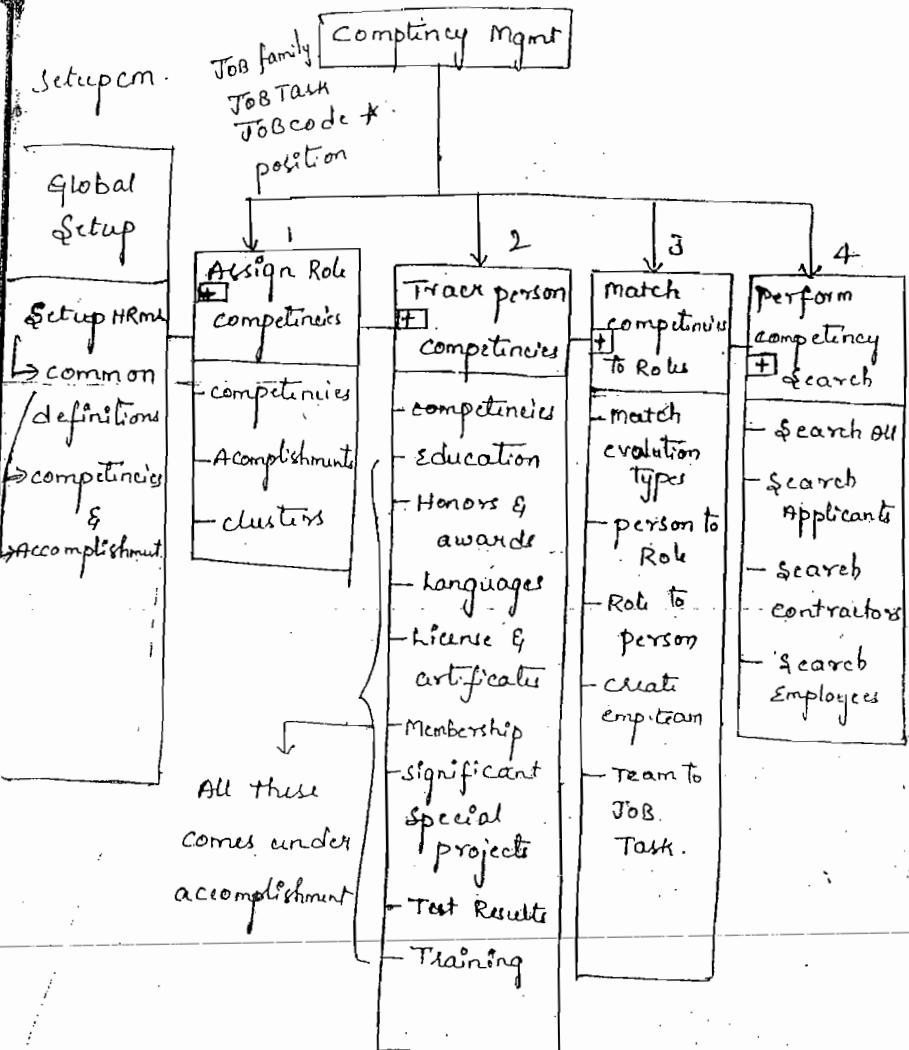
C

C

Navigation

Work

Report



### Navigations:

WORKFORCE Development → competency Management

↓  
- [+] Assign Role competencies

- [+] Track person "

- [+] Match competencies  
To Roles

- [+] Perform competency  
search

Report Manager

Role may be → JOB code \*

position → used for govt. organisation

JOB Family

JOB Task

→ who

①

②

③

→ Competency mgt is mainly used while recruiting.

### Base Benefits

we have diff. kinds of benefits

- |                 |                       |
|-----------------|-----------------------|
| ① Life Benefits | ④ Leave Benefits      |
| ② Health "      | ⑤ Retirement "        |
| ③ Saving "      | ⑥ Pension "           |
| ⑦ FSA           | ⑦ vacation Buy & Sell |

→ we can divide these benefits into 2 types

- ① mandatory — according to law, the cmp must offer the benefits.  
② voluntary — These benefits are mainly to retain the employees.

→ Who will give Benefits?

Vendors will give "

These " won't " " freely, for that we have to pay premium

→ who will pay premium?

Employer will pay — something

Employee " " — something

Sometimes combination of Employer & Employee will pay

### Sequence

① Defn

② Def

③ Assig

→ Each

④ Enr

⑤ calc

① To

⑤

① & ②

③ & ④

### Navigat

### Benefit

→ while calculating premium we do some calculations.

① Age Graded → Age ↑ premium ↓ depending on

② Fixed

③ Salary of employee.

### Sequence of steps to be followed for Base Benefits

① Define Benefit plans & other control Tables.

② Define Benefit programs.

③ Assign Benefit program to employees.

→ Each employee can have only one Benefit prgm.

④ Enroll Employee into the Benefit plans.

⑤ Calculate Benefit Deductions.

① to ④ steps comes under Base Benefits module.

⑤<sup>th</sup> step comes under payroll module.

① & ④ steps comes under setup specific to Base

Benefits

③ & ④ " " " Tx Basic Benefit.

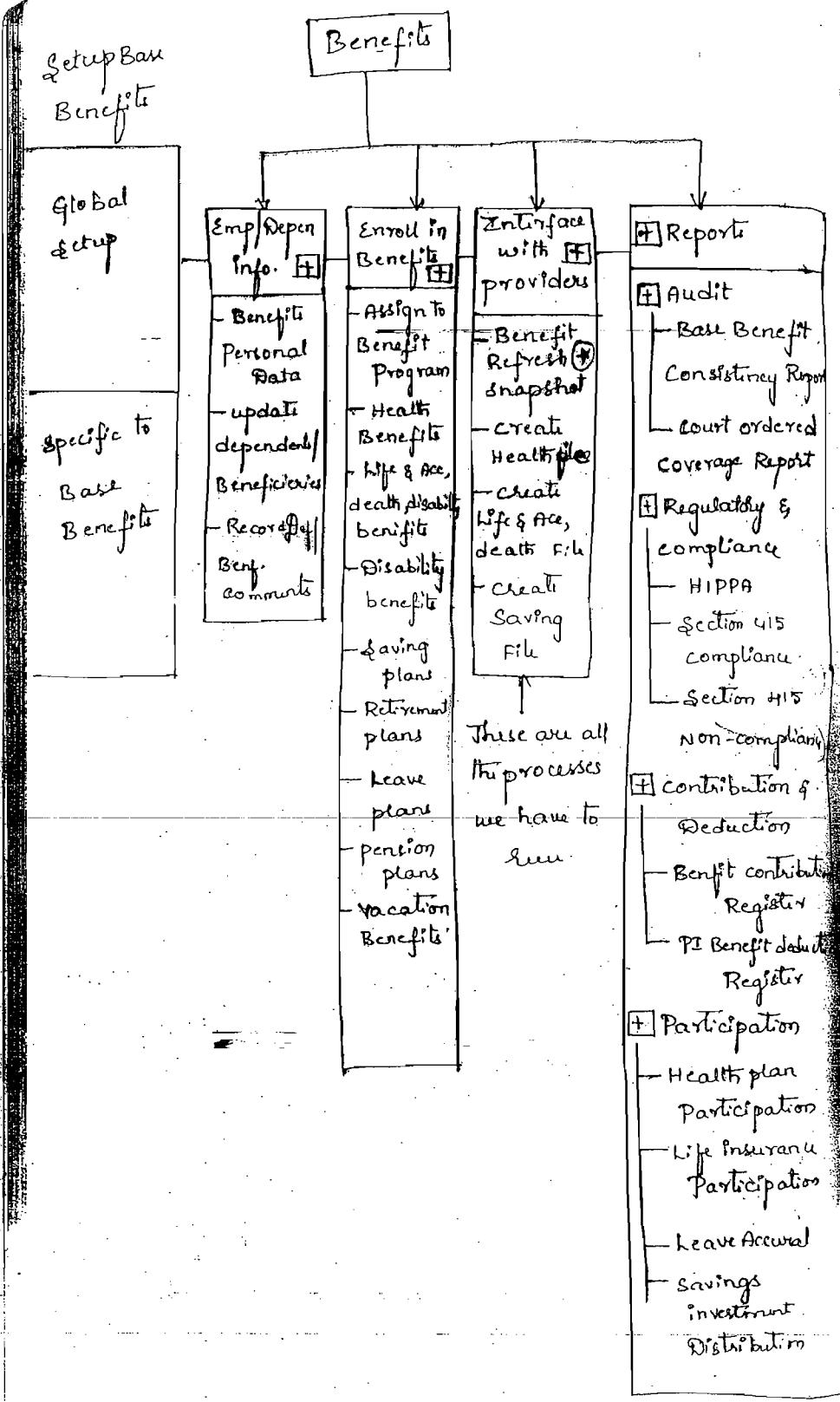
### Navigations:

Benefits →  Employee/Dependent Information

Enroll in Benefits

Interface with providers

Reports



Setup HRMS → product Related



Base Benefits → Benefit Plan Table.

→ If we want to change the Translate values.

PT → utilities → Administration.



Translate values.

Give the Field Name & mark Values as Inactive.

→ If we want to add a new plan, specify the range of plan to which it belongs.

0 - 9 → General  
10 - 19 → Health.

20 - 29 → Life.

Query Manager

30 - 39 → Disability.

40 - 49 → Savings.

50 - 59 → Leave.

60 - 69 → F.S.A (Flexible Spending Account)

70 - 79 → Retirement plans

80 - 89 → pension plans

90 - 99 → vacation.

Benefit\_PLAN\_TABLE - Comp

Benefits



Child benefit

Benefit\_PLAN\_TBL - Rec

↓  
Arr to Benefit Proj

PLAN\_TYPE } - R

Ben\_Pro\_Participant - Comp

Benefit\_Rate

Ben\_Pro\_Participant

GFFDC

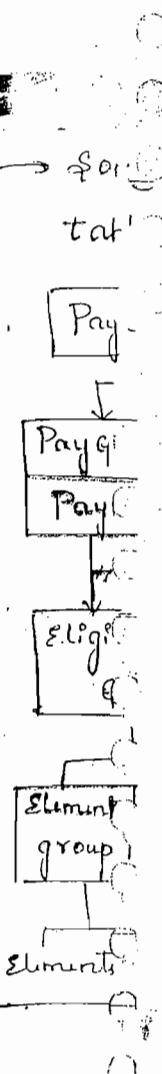
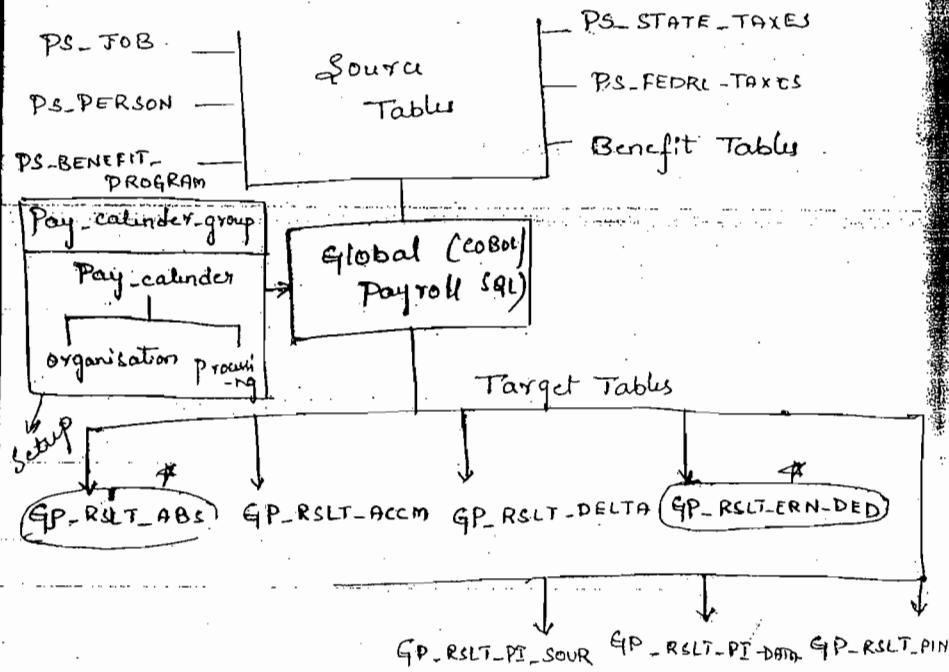
Emp - RCD (Sub Record)

Emp - Ben\_Sett - Settlement

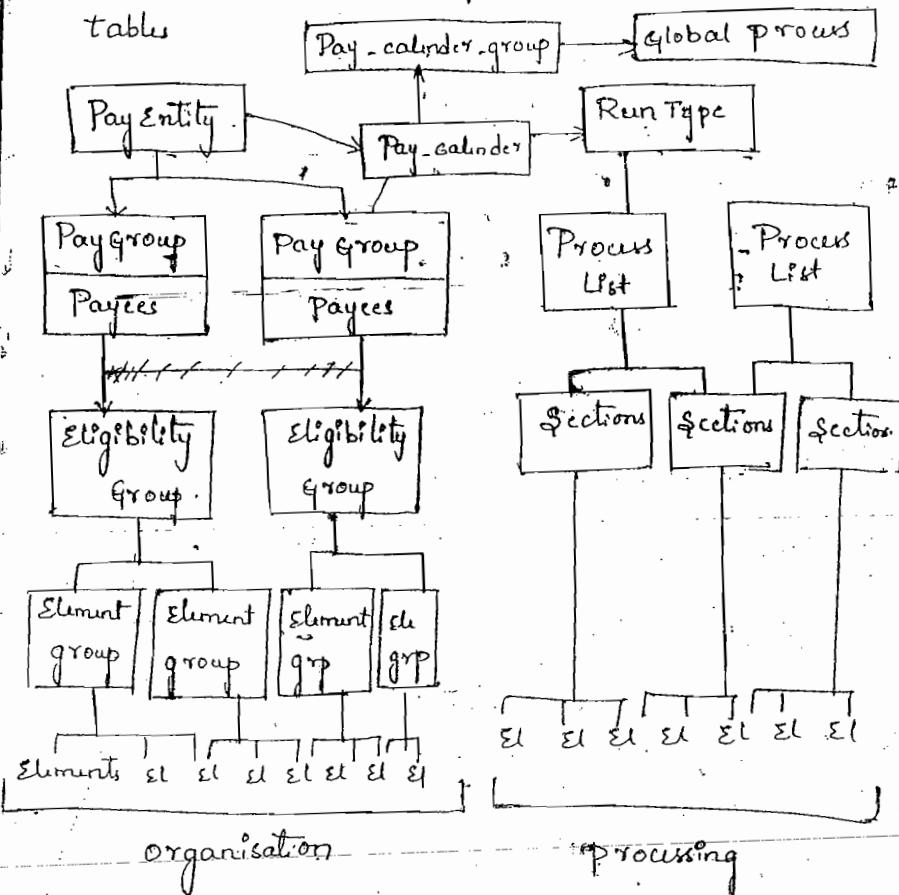
↓

## Global Payroll

- Global payroll is a process (or) Engine.
- " " takes data from source tables & populate them into target tables.
- Global payroll is empty engine.
- we define some rules to global payroll.
- All the rules we group under pay-calender-group.
- pay-calender-group have pay-calender.
- pay-calender rules are divided into
  - ① Organisational
  - ② Processing
- According to the rules defined payroll process the source tables & populate data into target table.



→ Four Tables are mainly Administer work force



Processing head — Run type

Organisation — Pay Entity

Navigations:

Global payroll

→  Payee Data

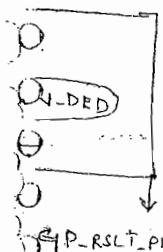
Absence mgmt

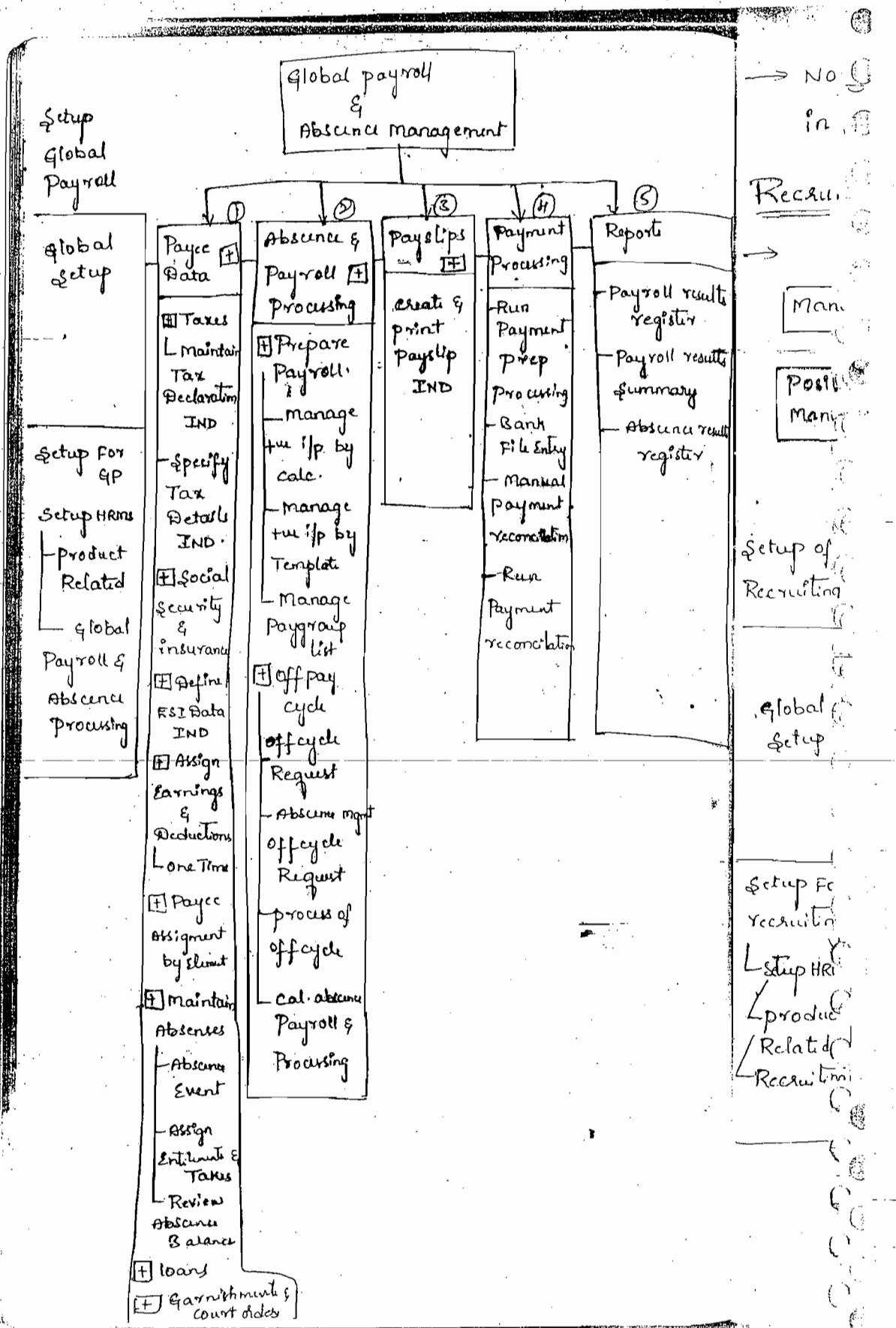
Absence & payroll processing

Payslips

Payment processing

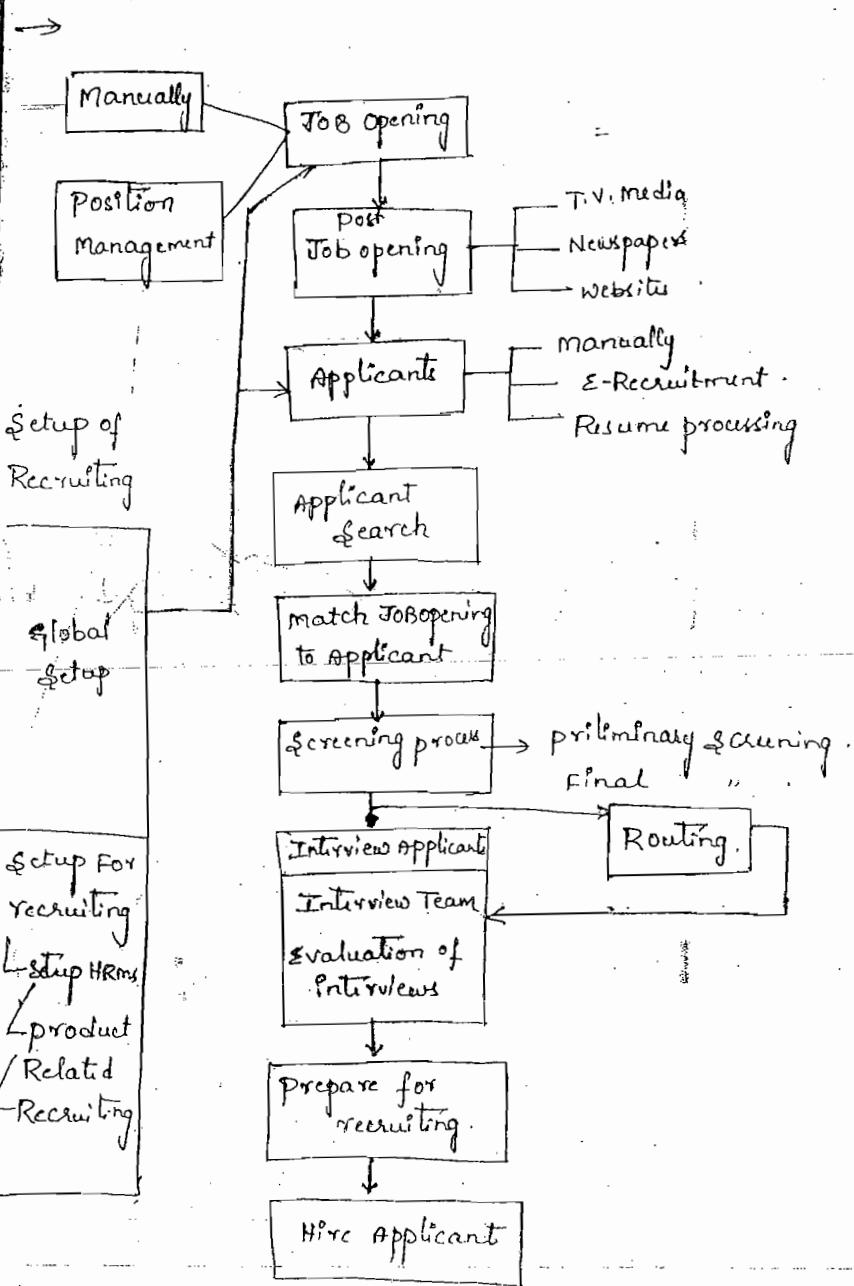
Reports





→ North American payroll all the rules are predefined in payroll process.

### Recruiting:



9502321202

VAnand@gnet

tv@catalogs.com