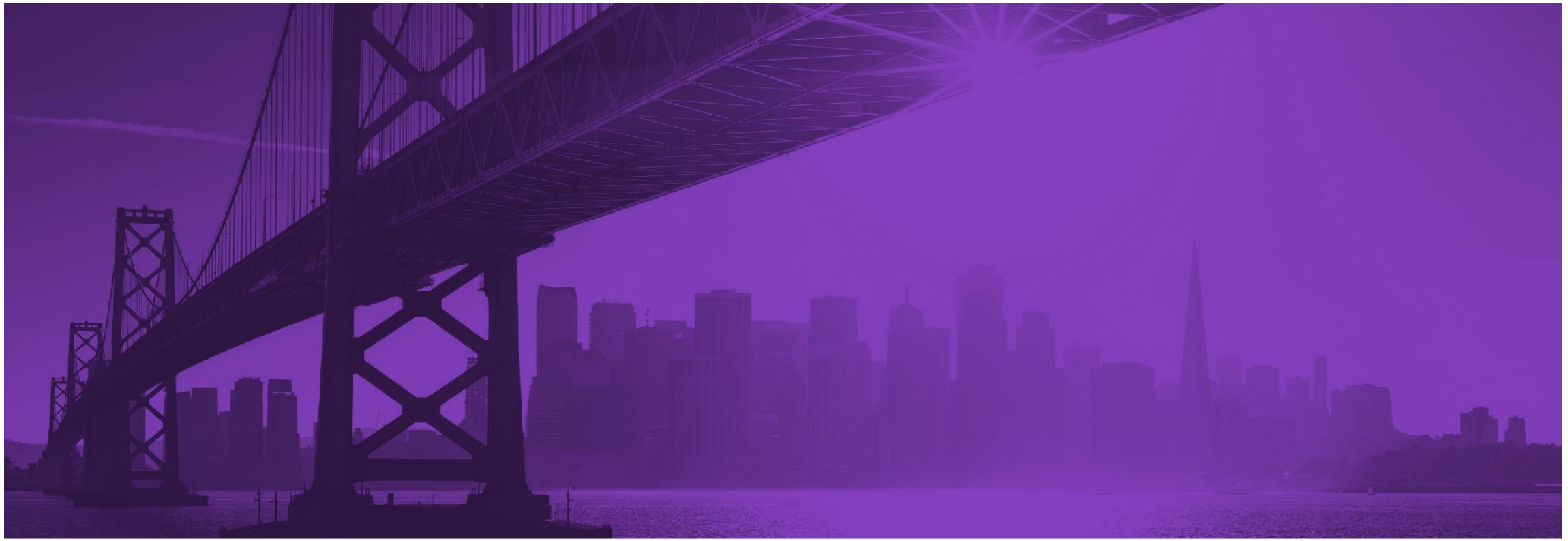


NN IZWS WAPL 101

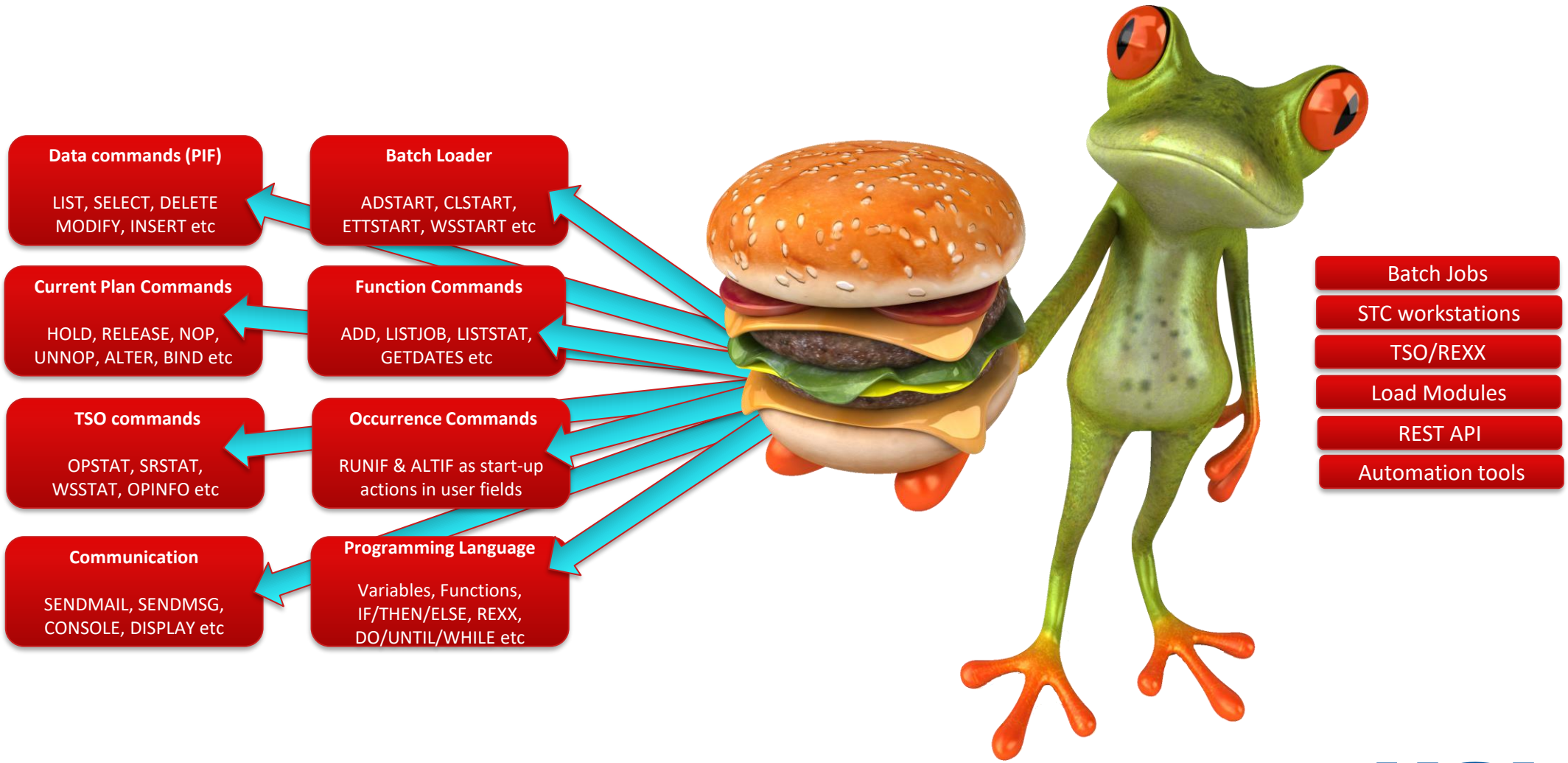
No more juggling frogs

What's with the frogs? I brief history of WAPL

- In the beginning there was EQQPIFOP
  - Great REXX sample from Doug Specht
  - Each time you wanted to use PIF you wrote a whole new program
- I did a presentation at ASAP in 2008 called Juggling Frogs
  - Comparing using PIF to Juggling Frogs
  - It was tricky, but you could learn to do it
- In 2009 Scheduling Operational Environment was released
  - A **F**ree **R**EXX **O**perational **G**oodybag of reusable PIF commands
  - Frogs Not Required
- In 2015 SOE gained programming capability and became WAPL
  - And became officially part of the product
  - For HCL z/OS Workload Automation, WAPL is the only PIF tool

So what is WAPL

- Made from many ingredients



Running WAPL from a Batch Job

- By far the simplest way to run WAPL is in batch

```
//RUNWAPL   EXEC EQQYXJPX,
//            SUBSYS=IWSX
//OUTBL     DD SYSOUT=*
//OUTDATA   DD SYSOUT=*
//SYSIN     DD *
SHOW OPTIONS
```

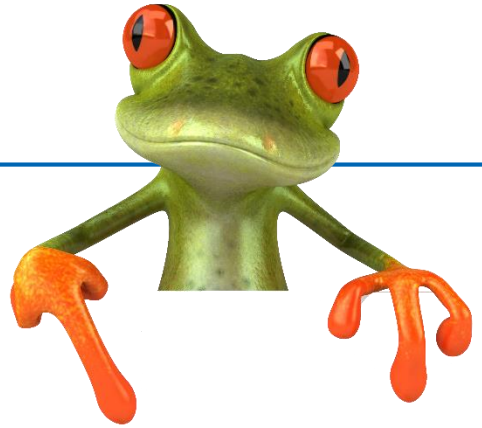
Proc to use from APAR PI79321

Controller name

Optional Batch Loader output

Optional ILSON output

New JCL please



- WAPL has a brand new bag (well proc)
- From PI79321 onwards we recommend the new **EQQYXJPX** procedure instead of EQQYXJCL
  - It removes EQQFILE, EQQREF and EQQLANG from the JCL, reducing I/O
  - EQQOPTS isn't there by default, but you can code your own
  - This make WAPL leaner and greener
- LOADDEF command replaces FILESPEC= in JCL
  - LOADDEF AD* loads definitions of all AD segments
  - LOADDEF ADCOM loads only ADCOM definition
- Both DATA (ILSON) and LOADER streams enabled automatically
  - You can turn off stream with an override
  - e.g. LOADDEF AD* DATA(-) defines full Batch Loader output but no ILSON data

Running WAPL from an STC workstation

- If your Systems Programmer sets up STC workstations you can fly

```
---------------------------- OPERATION USER FIELDS ----------- Row 1 to 2 of 2
Command ===>                                                  Scroll ===> CSR

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete

Application         : DH#WAPL                    WAPL Demo
Operation           : CMD1 001     Run WAPL commands
Jobname             : TWSXCMD1

Row  User Field Name  User Field Value
cmd                   ----+----1----+----2----+----3----+----4----+----5----
'''' EQQ-SYSIN-01     SHOW OPTIONS
'''' EQQ-SYSIN-02     SHOW VARIABLES
************** **************** Bottom of data ********** **** ***********
```

STC workstation

Standard Procedure/Job

Based on EQQWCMD1/2

User Fields with EQQ-SYSIN- prefix

You can do this even without STC workstations by coding this in the SYSIN of your job INCLUDE USER_FIELD(EQQ-SYSIN-*)

Exporting an application

- Simple batch loader job

```
//RUNWAPL   EXEC EQQYXJPX,
//          SUBSYS=IWSX
//OUTBL     DD SYSOUT=*
//SYSIN     DD *
OPTIONS STRIP(Y) SHOWDFLT(N)
LOADDEF AD* DATA(-)
LIST ADCOM ADID(DH#*) VALID(=) SELECT(Y)
```

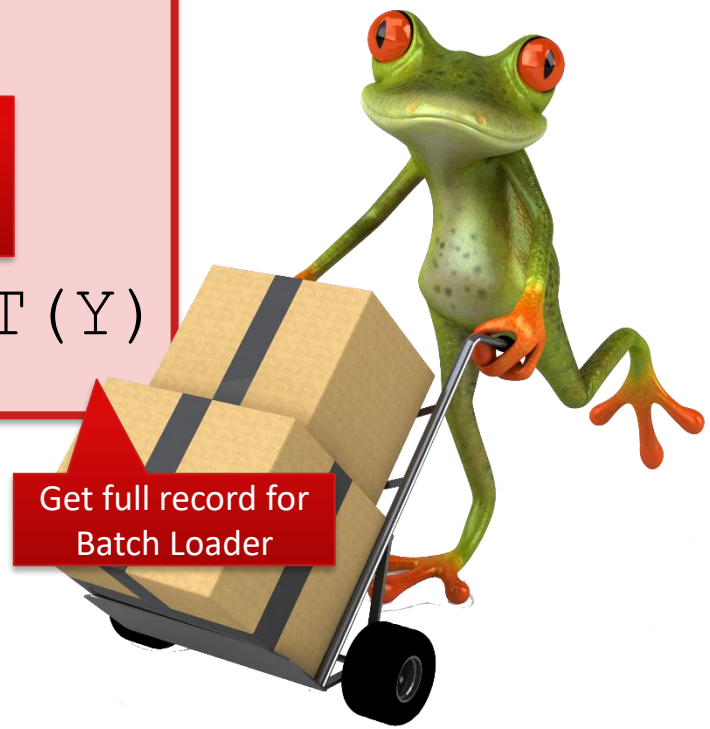
Generate minimum output

Loads the OUTPUT statements for Applications

LIST allows wildcards

Pick today's version

Get full record for Batch Loader



Objects, Records and segments

- There are many types of object within the ZWS database
  - e.g. Workstations, Calendars, Periods, Applications
- Each instance of an object is stored in a record
  - Each record is split into segments

ADCOM Segment

```
------------------------- MODIFYING AN APPLICATION --------------------------
Command ===>

Enter/Change data below:
Enter the RUN command to select run cycles, the DEP command to select
dependencies at application level or the OPER command to select operations.

Application id          : DH#FIRSTFRI13
Valid from - to         : 18/11/15 - 71/12/31

APPLICATION TEXT    ===> _________________________  Descriptive text
TYPE                ===> A          A = Application, G = Group definition
OWNER ID            ===> DINO_____________
OWNER TEXT          ===> _________________________
PRIORITY            ===> 5          A digit 1 to 9 , 1=low, 8=high, 9=urgent
VALID FROM          ===> 18/11/15   Date in the format YY/MM/DD
STATUS              ===> A          A - Active, P - Pending
AUTHORITY GROUP ID  ===> ________   Authorization group ID
CALENDAR ID         ===> ________________  For calculation of work and free days
GROUP DEFINITION    ===> ________________  Group definition id
SMOOTHING FACTOR    ===> ___    LIMIT  ===> ___     Deadline Feedback options

Last updated by DEAN     on 19/10/28 at 15.47
```

```
ADCOM -+- Common segment (1 per appl)
|
+= ADRUN =+= Run Cycle(s)
|         |
|         +- ADRULE - Rule (1 per run cycle)
|
+= ADAPD = Application dependencies
|
+= ADOP =+= Operation(s)
         |
         += ADDEP = Dependency(ies)
         |
         += ADXIV = External dependency interval(s)
         |
         += ADSR = Special resource(s)
         |
         +- ADOPEXT - Extended name (1 per op)
         |
         +- ADOPSAI - System Automation (1 per op)
         |
         += ADCNC = Condition(s)
         |
         += ADCNS = Conditional dependency(ies)
         |
         += ADCIV = Conditional dependency interval(s)
         |
         += ADUSRF = User field(s)
         |
         += ADVDD = Variable duration(s)
         |
         +- ADRE =- Remote job (1 per op)
```

Objects, Records and segments

- There are many types of object within the ZWS database
  - e.g. Workstations, Calendars, Periods, Applications
- Each instance of an object is stored in a record
  - Each record is split into segments

ADRUN Segments

```
-------------------------------- RUN CYCLES ----------------- Row 1 to 2 of 2
Command ===>                                                 Scroll ===> CSR

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Specify run days/Modify rule

Application            : DH#FIRSTFRI13
     Name of rg/                              In       Out of
Row  period/rule Input Deadline       F day effect   Effect   Variable table
cmd              HH.MM day HH.MM Type rule  YY/MM/DD YY/MM/DD
'''' FRI1AND3    12.00 01  13.00 R    4     72/01/01 72/01/02 ________________
     Text : ____________________________________________________
     Shift: ___0  Shift Day Type: _

'''' FRI1AND2    12.00 01  13.00 R    4     72/01/01 71/12/31 ________________
     Text : ____________________________________________________
     Shift: ___0  Shift Day Type: _

****************************** Bottom of data ******************************
```

```
ADCOM -+- Common segment (1 per appl)
|
+= ADRUN =+= Run Cycle(s)
|         |
|         +- ADRULE - Rule (1 per run cycle)
|
+= ADAPD = Application dependencies
|
+= ADOP =+= Operation(s)
         |
         += ADDEP = Dependency(ies)
         |
         += ADXIV = External dependency interval(s)
         |
         += ADSR = Special resource(s)
         |
         +- ADOPEXT - Extended name (1 per op)
         |
         +- ADOPSAI - System Automation (1 per op)
         |
         += ADCNC = Condition(s)
         |
         += ADCNS = Conditional dependency(ies)
         |
         += ADCIV = Conditional dependency interval(s)
         |
         += ADUSRF = User field(s)
         |
         += ADVDD = Variable duration(s)
         |
         +- ADRE =- Remote job (1 per op)
```

Objects, Records and segments

- There are many types of object within the ZWS database
  - e.g. Workstations, Calendars, Periods, Applications
- Each instance of an object is stored in a record
  - Each record is split into segments

ADRULE Segment

```
------------------------------ MODIFYING A RULE ------------------------------
Command ===>

Enter the GENDAYS command to display the dates generated by this rule
Enter the E command to specify EVERY options
Enter S and user data in the fields below to define a rule

Application   : DH#FIRSTFRI13
Rule          : FRI1AND2

 --- Frequency ---          --- Day ---     --- Cycle Specification ---
--------------------------------------------------------------------------------
        _ Only          |  S Day        |  _ Week       _ January     _ July
        S Every         |  _ Free day   |  S Month      _ February    _ August
                        |  _ Work day   |  _ Year       _ March       _ September
 _ First      _ Last    |  _ Monday     |               _ April       _ October
 _ Second     _ 2nd Last|  _ Tuesday    |               _ May         _ November
 _ Third      _ 3rd Last|  _ Wednesday  |               _ June        _ December
 _ Fourth     _ 4th Last|  _ Thursday   |  Week number  __ __ __ __ __ __
 _ Fifth      _ 5th Last|  _ Friday     |  Period/RG    ________ ________
   ___ ___      ___ ___ |  _ Saturday   |  name         ________ ________
   ___ ___      ___ ___ |  _ Sunday     |
   ___ ___      ___ ___ |               |  Shift default origin by  ___ days
--------------------------------------------------------------------------------
```

```
ADCOM -+- Common segment (1 per appl)
|
+= ADRUN =+= Run Cycle(s)
|         |
|         +- ADRULE - Rule (1 per run cycle)
|
+= ADAPD = Application dependencies
|
+= ADOP =+= Operation(s)
         |
         += ADDEP = Dependency(ies)
         |
         += ADXIV = External dependency interval(s)
         |
         += ADSR = Special resource(s)
         |
         +- ADOPEXT - Extended name (1 per op)
         |
         +- ADOPSAI - System Automation (1 per op)
         |
         += ADCNC = Condition(s)
         |
         += ADCNS = Conditional dependency(ies)
         |
         += ADCIV = Conditional dependency interval(s)
         |
         += ADUSRF = User field(s)
         |
         += ADVDD = Variable duration(s)
         |
         +- ADRE =- Remote job (1 per op)
```

Objects, Records and segments

- There are many types of object within the ZWS database
  - e.g. Workstations, Calendars, Periods, Applications
- Each instance of an object is stored in a record
  - Each record is split into segments

ADOP Segments

```
-------------------------------- OPERATIONS ----------------- Row 1 to 1 of 1
Command ===>                                                 Scroll ===> CSR

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Select operation details, J - Edit JCL
Enter the TEXT command above to include operation text, or,
enter the GRAPH command to view the list graphically.

Application            : DH#FIRSTFRI13

Row  Oper      Duration  Job name  Internal predecessors       Morepreds  No.of
cmd  ws   no.  HH.MM.SS                                       -IntExt-   Conds
'''' NONR 001  00.00.01  ZFIRST__  ___ ___ ___ ___ ___ ___ ___  0   0      0
****************************** Bottom of data ********************************
```

```
ADCOM -+- Common segment (1 per appl)
|
+= ADRUN =+= Run Cycle(s)
|         |
|         +- ADRULE - Rule (1 per run cycle)
|
+= ADAPD = Application dependencies
|
+= ADOP =+= Operation(s)
         |
         += ADDEP = Dependency(ies)
         |
         += ADXIV = External dependency interval(s)
         |
         += ADSR = Special resource(s)
         |
         +- ADOPEXT - Extended name (1 per op)
         |
         +- ADOPSAI - System Automation (1 per op)
         |
         += ADCNC = Condition(s)
         |
         += ADCNS = Conditional dependency(ies)
         |
         += ADCIV = Conditional dependency interval(s)
         |
         += ADUSRF = User field(s)
         |
         += ADVDD = Variable duration(s)
         |
         +- ADRE =- Remote job (1 per op)
```

Objects, Records and segments

- There are many types of object within the ZWS database
  - e.g. Workstations, Calendars, Periods, Applications
- Each instance of an object is stored in a record
  - Each record is split into segments

ADDEP

ADVDD

ADOPSAI

ADRE

ADSR

ADEXT

ADUSRF

```
------------------------------ OPERATION DETAILS ----------------------------
Option ===>

Select one of the following:

                                                                More:     +
  1 PREDECESSORS            - List of predecessors
  2 WS RES AND SERVERS      - Work station resources and servers
  3 SPECIAL RESOURCES       - List of special resources
  4 AUTOMATIC OPTIONS       - Job, WTO, and print options
  5 FEEDBACK                - Feedback options
  6 TIME & RUN CYCLE OPT.   - Time and run cycle options specifications
  7 OP INSTRUCTIONS         - Operator instructions
  8 JCL EDIT                - Edit JCL
  9 CLEANUP OPTIONS         - Cleanup Options
 10 EXTENDED INFO           - Operation extended info
 11 AUTOMATION INFO         - System Automation operation info
 12 USER FIELDS             - User Fields operation info
 13 REMOTE JOB INFO         - Remote job information

 Application            : DH#FIRSTFRI13
 Operation              : NONR 001
 Jobname                : ZFIRST          Number of int preds    : 0
 Duration               : 00.00.01        Number of ext preds    : 0
                                          Number of conditions   : 0
```

```
ADCOM -+- Common segment (1 per appl)
|
+= ADRUN =+= Run Cycle(s)
|         |
|         +- ADRULE - Rule (1 per run cycle)
|
+= ADAPD = Application dependencies
|
+= ADOP =+= Operation(s)
         |
        += ADDEP = Dependency(ies)
         |
        += ADXIV = External dependency interval(s)
         |
        += ADSR = Special resource(s)
         |
        +- ADOPEXT - Extended name (1 per op)
         |
        +- ADOPSAI - System Automation (1 per op)
         |
        += ADCNC = Condition(s)
         |
        += ADCNS = Conditional dependency(ies)
         |
        += ADCIV = Conditional dependency interval(s)
         |
        += ADUSRF = User field(s)
         |
        += ADVDD = Variable duration(s)
         |
        +- ADRE =- Remote job (1 per op)
```

LIST vs SELECT

- The two most common PIF commands to get data from the database or plans
- LIST – Search for items in the database
  - Can search using keywords and wildcards
  - Can return multiple objects
  - Only returns the common segment
- SELECT – Retrieve a specific records
  - Keywords must explicitly identify one object
  - Can only return one object
  - Returns the whole record

- For best of both worlds LIST with SELECT(Y) as a keyword
  - This can find and return multiple whole records

How to generate output from LIST and SELECT

- Two types
  - Batch Loader – Reproduces and object in text form
  - ISPF Loader Streamed Output Notation (ILSON) – Export data segment by segment, field by field
- An OUTPUT statement tells WAPL which segments and field to export
  - You can write your own by hand
  - You can INCLUDE pre-prepared full segment members from SEQQMISC
  - The new LOADDEF command can select full segments quickly and easily
- To generate Batch Loader you must SELECT the entire record
  - LIST ADCOM ADID(MYAPPL) SELECT(Y)
- For ILSON data you can LIST or SELECT

What is Batch Loader?

- Basic batch loader looks like this –

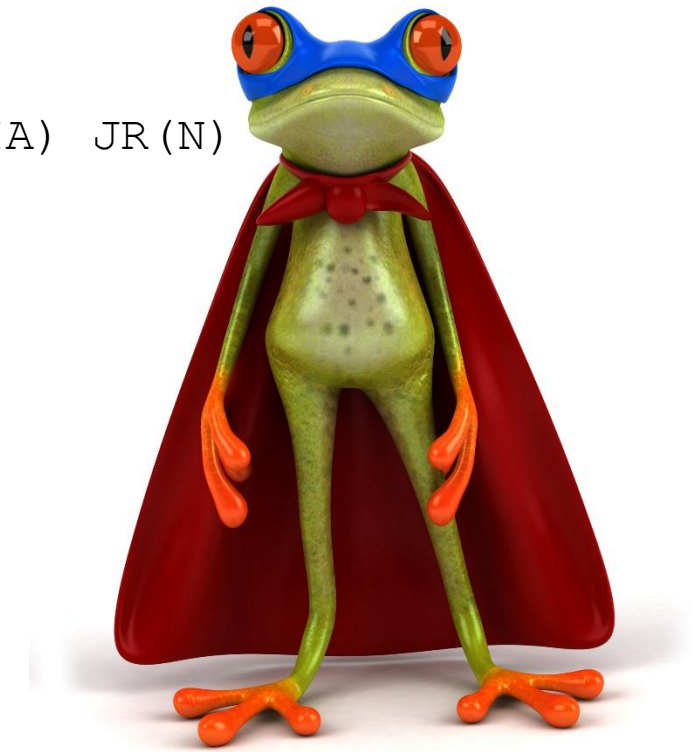
```
ADSTART ADID(DH#FIRSTFRI13) ADVALFROM(181115) OWNER(DINO)
ADRUN SEQ(00000001) NAME(FRI1AND3) VALFROM(720101) VALTO(720101)
    IATIME(1200) DLDAY(1) DLTIME(1300)
ADRULE ONLY(001 015) DAY(DAY) PERIOD(FIRSTFRI)
ADOP WSID(NONR) OPNO(001) JOBN(ZFIRST) DURATION(1)
```

- This is native WAPL language and can be used as input to a further WAPL step
- OPTIONS DBMODE(<mode>) allows you to choose how to process the statements
  - ADD/REPLACE – Create or replace an entire object from the statements
  - UPDATE/COPY – Change or copy an existing object, just specifying the differences
  - EXPORT – Generate translated Batch Loader
  - SCAN – Basic syntax check

Not just Applications

- Special resources
  - SRSTART RESNAME(DH#.TEST.RES1) GROUP() HIPER(Y) USEDFOR(B) ONERROR()
    DESCR() ONCOMPLETE() MAXTYPE(R) MAXLIMIT(0) QUANTITY(1) AVAIL(Y)
    SRDWS WSID(*)
- Event Triggers
  - ETTSTART ETTTYPE(R) ETTNAME(TRIG.TXFB#OPIA) ADID(TXFB#OPIA) JR(N)
    DR(Y) AS(N)
- Get them all with EXPAND

```
//RUNWAPL  EXEC EQQYXJPX,
//         SUBSYS=TWSX
//OUTBL    DD SYSOUT=*
//OUTBL2   DD SYSOUT=*
//SYSIN    DD *
OPTIONS STRIP(Y) SHOWDFLT(N) EXPAND(Y)
LOADDEF * DATA(-)
LOADDEF AD* DATA(-) LOADER(OUTBL2)
LIST ADCOM ADID(*) SELECT(Y)
```



Truly dynamic applications

- Batch loader can create an application from scratch
  - Directly into the current plan
  - Use ACTION(SUBMIT) on the ADSTART keyword
- Batch loader enhanced to make it easy to create workload on the fly
  - AUTOPRED/AUTOSUCC makes it easy to create linear or parallel flow

```
VARSUB SCAN
ADSTART ACTION(SETDEFAULT)
   ADOP OPNO(005) DURATION(1)
ADSTART ACTION(SUBMIT) ADID(DYNAPPL!CHHMM) OWNER(DEAN) GROUP(ADCDMST)
   ADOP WSID(NONR) OPNO(001) JOBN(ZFIRST) AUTOPRED(PREV)
   ADOP WSID(CPU1) JOBN(JOB005)
   ADOP WSID(CPU1) JOBN(JOB010)
   ADOP WSID(CPU1) JOBN(JOB015)
   ADOP WSID(CPU1) JOBN(JOB020)
   ADOP WSID(NONR) OPNO(255) JOBN(ZLAST)
```

Sets the default for all operations

Makes all following operations dependent on the previous

Lifecycle management

- The TRANSLATE command can define rules to translate element names in Batch Loader output
  - The TRANSLATE command should precede any LIST statements

```
TRANSLATE AD FILTER(TEST*) OVERLAY(PROD*)
```

```
TRANSLATE JS FILTER(Z*) OVERLAY(Z*)
             FILTER(T*) OVERLAY(P*)
```

```
TRANSLATE WS OLD(TEST) NEW(PROD)
             OLD(CPUT) NEW(CPUP)
```

- Any batch loader created by LIST after these TRANSLATE statements will be transformed to apply to your next environment
- Batch loader can be TRANSLATED using OPTIONS DBMODE(EXPORT)

UPDATE mode

- In some cases you may want to change a single part of an object
  - Specifying the full object can be a bit of a bind
  - In some cases you might not “know” the whole object
- Variable tables are a classic example of this
  - Updating a variable individually as part of your workload
  - Makes it difficult to “know” the full content
  - You would need to unload the table, change the variable and reload
- Fortunately DBMODE(UPDATE) does this for you

```
OPTIONS DBMODE(UPDATE)
JCLVSTART JCLVTAB(TESTTAB)
JCLVVAR VARNAME(ZLP) DEFAULT(X)
```

Identify table

Identify table

Only specify keywords you want to change

Current Plan commands – PIF based

- You can script almost anything in the Current Plan using a set of PIF based commands
  - INSERT CPOC – Add an occurrence to the plan
  - MODIFY CPOC/CPOP – Modify occurrences or operations
  - DELETE CPOP/CPOP – Delete occurrences or operations
  - Plus many more variants
- These can be used in combination to achieve complex operations

```
INSERT CPOC ADID(MYAPPL)
MODIFY CPOP OPNO(005) JOBNAME(NEWJOB)
```

- If you can do it through the 5.2 or 5.3 panels, you can probably do it through WAPL
  - Clean-up and restart being the main exception

Current Plan Operation Commands

- A whole set of commands, all work the same way
- ALTER, BIND, FIND, FORCE, HOLD, KILL, NOP, QUEUE_BEHIND, RELEASE, REPLY, UNNOP
- Simple usage like this – HOLD MYJOB
  - Will look for the earliest occurrence of MYJOB in the CP in a status that hasn't run yet
- Add COUNT(0) to the command and it will hold all matching jobs
- Wildcards can be used – HOLD MY* COUNT(0)
- Scope can be limited with keywords like DATE, TIME and RANGE
- Many other keywords can be used to target each command

And so much more

- WAPL has variables
  - These can be used internally for programming
  - They can be saved externally into JCL variable tables
  - They can represent an entire object in detail
- It is a full programming language
  - IF/THEN structures
  - DO loops
- Many other useful commands
  - Send console messages and emails
  - Dynamic interval processing
  - Job status checking

Any questions?

Please submit your session feedback!

- Do it online at http://conferences.gse.org.uk/2019/feedback/nn

1. What is your conference registration number?

This is the three digit number on the bottom of your delegate badge

2. Was the length of this presention correct?

1 to 4 = "Too Short" 5 = "OK" 6-9 = "Too Long"

1 2 3 4 5 6 7 8 9

3. Did this presention meet your requirements?

1 to 4 = "No" 5 = "OK" 6-9 = "Yes"

1 2 3 4 5 6 7 8 9

4. Was the session content what you expected?

1 to 4 = "No" 5 = "OK" 6-9 = "Yes"

1 2 3 4 5 6 7 8 9

Session NN

https://www.ibm.com/developerworks/community/groups/community/zGlue