Week - 2 Reading – RDp, RPG & DDS

- Rational Developer for Power Systems (RDp)
- Introduction to RPG (Report Program Generator Language)
- Introduction to DDS (Data Description Specifications)

Rational Developer for Power Systems (RDp)

All great software usually evolves from little bits of code that programmers write to make their jobs a little easier. Rational Developer for Power Systems (RDp) is no exception. As RDp boots up, you'll see a message that the software was based on Eclipse technology.

Eclipse is a software development environment created by IBM. It started out as a collection of tools that IBM programmers had written and shared amongst themselves. The tools were mostly written for Java development but could be applied to other languages such as C++ and xHtml. Eclipse impressed everyone at IBM, including the iSeries Development team. The iSeries team evolved Eclipse into an iSeries Development environment and called it Websphere! The first versions of Websphere were hard to install and left a huge footprint on computers. The screen design tool, Code Designer was a separate tool which lead to lots of file locking issues.

RDp resolved a lot of issues from Websphere. The new Screen Designer is fully integrated into the software and the footprint is much smaller than the original Websphere. IBM no longer supports the green screen Program Development Manager with the hopes that everyone will convert to RDp. Unfortunately, it's tough to teach old programmers new tricks!

RPG (Report Program Generator) ^

RPG is a registered trademark by IBM and it does not stand for Role Playing Games!

The language was originally developed with Accountants in mind, to replace the tedious job of adding up large volumes of numbers. The language is full of short cuts, making each line of code very powerful. An 8 line RPG report program will take hundreds of lines in COBOL. Programmers love it in the IBM i world because it's less to type! Remember, back in the old days of computing, a line of code was a card about the size of a business envelope. 8 cards was a lot easy to manage, then hundreds!

An RPG program usually starts with File Specification lines where the files used by the program are defined. FSpecs are often followed by DSpecs (Data Definitions) where other variables needed by the program are defined. These two types of lines still use the fixed format column specific syntax required back in the old days. With the new version of RPG, we now have free-format C Specs (Calculation Specifications) and now RPG looks a lot like C. Just remember, every line must end in a semi-colon.

DDS (Data Description Specifications) ^

We use DDS to define data. Remember that our definition of data on this server includes screen designs as we'll see in Lab 3.

The database, DB2 is built into ibm i. DB2 is simple in concept and consists of physical files (also called tables) and logical files (also called indexes or views). You can use SQL of DDS to maintain your databases. Either is equally valid. Just remember, once you maintain an object using SQL, you must continue to SQL as your DDS code is out of date. SQL is used more often than DDS in the market place, but it's important to understand the history in order to appreciate the present.

DDS is a case sensitive language. Everything must be typed in uppercase. The language is also very column specific, so it's a good idea to use the source prompters when maintaining code.

DDS programs start out with file level keywords, or attributes that apply to a file itself. After that, a record format is listed. Record format means a layout or a screen. Physical files only have one record format, but display files have one record format for each different screen layout. The fields are then listed, followed by the access path (key) information.

Fields can have the following data types:

Entry	Meaning
P	Packed Decimal
S	Zoned Decimal
B	Binary
F	Floating-Point
A	Character
H	Hexadecimal
L	Date
T	Time
Z	TimeStamp

You can define the same size of number in either both Packed or Zoned Decimal format. A Packed Decimal takes up less space in memory than a Zoned Decimal. A Packed Decimal stores two digits in one byte. A number that has a 999,999.99 as it's largest possible value would take up 8 bytes as a Zoned Decimal and 5 bytes as Packed Decimal. If you leave the number of decimal places blank, then none are assume. If you leave the data type blank, then character is assumed.

We don't often see Binary, Floating-Point and Hexadecimal fields in business applications.

```
T.Name++++++RLen++TDp.......Functions+++++++++++++++++++++++++++++++++++
_ __________ _____ __       UNIQUE___________________
R ACCTPFR___ _____ __       TEXT('ACCOUNT INFORMATION')__
_ ACCT   ___ ____3S_0       COLHDG('ACCOUNT'  'NUMBER')____
_ ACCTNAME _ ___30A__       COLHDG('ACCOUNT'  'NAME')____
_ ACCTDATE__ _____L__       COLHDG('ACCOUNT'  'DATE')____
_ AMTOUT   __ ____7P_2       COLHDG('AMOUNT'  'OWING')____
K ACCT____ _____ __       _____________________________
```

The above DDS code will create a physical file to store Account Information. The record format name is ACCTPFR which means that the name of the physical file must be something other than ACCTPFR.

The file has 4 fields:

Field Name	Attributes	Column Headings
ACCT	This is a numeric field with a length of 3 digits. The field will take up 3 bytes in memory	Account Number
ACCTNAME	This is a 30 byte character field.	Account Name
ACCTDATE	This is date field. Data entered must be a valid date and must be entered in *ISO format (yyyy-mm-dd)	Account Date
AMTOUT	This is a 7 digit including 2 decimal places number, so the biggest possible value is 99,999.99. The field is packed, so it's stored in 4 bytes.	Amount Owing

The file has a key field of ACCT which means that the file has an access path which is sorted by ACCT.