|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  | LOAD FROM FILE : Load RT Integrated LMP    Datamodel  Populates the table MC\_PRICE\_RT.  Uses lookup tables used: MC\_PRICE\_SOURCE\_CODE\_TYPE, X\_XP and MC\_PRICE\_RT\_TYPE.    Outputs  <env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">  <env:Header/>  <env:Body>  <QueryResponse xmlns="http://markets.midwestiso.org/dart/xml"> n  <RealTimeLMP type = "xxxx" day= "yyyy-mm-dd">  <PricingNode location= "xxx">  <PricingNodeInterval interval= "yyyy-mm-ddThh:mm:ss">  <LMP>99.99</LMP>  <MCC>99.99</MCC>  <MLC>99.99</MLC>  <RegMCP>99.99</RegMCP>  <RegMileageMCP>99.99</RegMileageMCP>  <SpinMCP>99.99</SpinMCP>  <SuppMCP>99.99</SuppMCP>  </PricingNodeInterval>  </PricingNode>  </RealTimeLMP>  </QueryResponse>  </env:Body>  </env:Envelope>    Description     |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **Field**  **MC\_PRICE\_RT** | **UK** | **REQ** | **Description** | **Source** | **Lookup/Notes** | | XP\_ID |  | Y | Specifies the pricing location name. | Location= "xxx" | XP\_ID Where ISO\_REF = XXX | | START\_DT |  | Y | Indicates effective Start date of the record | interval= "yyyy-mm-ddThh:mm:ss" | This is the interval ending so must be converted to the beginning of the interval by subtracting 5 minutes. | | DT\_INTRVL\_VAL |  | Y | The interval value to denote the duration of the data. | +00 00:05:00.000000 | 5 minute interval data | | PRICE\_TYPE\_CD |  | Y | Price Type Code  Unique price types are driven by the tags in the file. If all tags exist and are populated then 5 physical records will be created in the table each with a different price type. |  | MC\_PRICE\_RT\_TYPE  <LMP> = 5MINRTLMP  <RegMCP> = 5MINRTREGMCP  <RegMileageMCP> = 5MINRTREGMILMCP  <SpinMCP> = 5MINRTSPINMCP  <SuppMCP> = 5MINRTSUPPMCP | | SOURCE\_CD |  | Y | Source Code | type = "xxxx" | MC\_PRICE\_SOURCE\_CODE\_TYPE  If Type = “ExAnte” the SOURCE\_CD = 'ISOEXA'  Else if Type = “ExPost” then SOURCE\_CD = 'ISOEXP' | | PRICE\_AMT |  |  | Specifies the hourly integrated real time LMP in $ per MW.  One physical record created for each populated tag. | <LMP>  <RegMCP><RegMileageMCP> <SpinMCP>  <SuppMCP> |  | | ENERGY\_AMT |  |  | Specifies the hourly integrated real-time energy component in $ per MW. The value is calculated when loaded. LMP – MCC – MLC = Energy. | LMP – MCC – MLC | Only populated for 5MINRTLMP record | | CONGEST\_AMT |  |  | Specifies the hourly integrated real-time congestion component in $ per MW. | <MCC> | Only populated for 5MINRTLMP record | | LOSS\_AMT |  |  | Specifies the hourly integrated real-time loss component in $ per MW. | <MLC> | Only populated for 5MINRTLMP record | | USER\_ID |  | Y | Specifies the user who loaded the record |  |  | | REVISED\_DT |  | Y | Specifies the system date when the record was last loaded |  |  |     Processing (Wipe and Load)  If price for given day / price type/ source DOES NOT exist, INSERT the new record.  If prices for a given day/ price type/ source ALREADY exists, DELETE the data for the day and INSERT the new records.  Price type = '5MINRTLMP', '5MINRTREGMCP', '5MINRTREGMILMCP', '5MINRTSPINMCP', '5MINRTSUPPMCP'  Examples    <env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">  <env:Header/>  <env:Body>  <QueryResponse xmlns="http://markets.midwestiso.org/dart/xml">  <RealTimeLMP type= "ExAnte" day= "2007-06-01">  <PricingNode location= "ABC">  <PricingNodeInterval interval= "2007-06-01 01:05:00">  <LMP>35.00</LMP>  <MCC>10.00</MCC>  <MLC>15.00</MLC>  <RegMCP>11.00</RegMCP>  <RegMileageMCP>5.00</RegMileageMCP>  <SpinMCP>12.00</SpinMCP>  <SuppMCP>13.00</SuppMCP>  </PricingNodeInterval>  </PricingNode>  <PricingNode location="ABC">  <PricingNodeInterval interval= "2007-06-01 01:10:00">  <LMP>45.00</LMP>  <MCC>11.00</MCC>  <MLC>12.00</MLC>  <RegMCP>14.00</RegMCP>  <RegMileageMCP>6.00</RegMileageMCP>  <SpinMCP>15.00</SpinMCP>  <SuppMCP>16.00</SuppMCP>  </PricingNodeInterval>  </PricingNode>  <PricingNode location="DEF">  <PricingNodeInterval interval= "2007-06-01 01:05:00">  <LMP>30.00</LMP>  <MCC>5.00</MCC>  <MLC>10.00</MLC>  <RegMCP>11.00</RegMCP>  <RegMileageMCP>5.00</RegMileageMCP>  <SpinMCP>12.00</SpinMCP>  <SuppMCP>13.00</SuppMCP>  </PricingNodeInterval>  </PricingNode>  <PricingNode location="DEF">  <PricingNodeInterval interval= "2007-06-01 01:10:00">  <LMP>27.00</LMP>  <MCC>7.00</MCC>  <MLC>5.00</MLC>  <RegMCP>14.00</RegMCP>  <RegMileageMCP>6.00</RegMileageMCP>  <SpinMCP>15.00</SpinMCP>  <SuppMCP>16.00</SuppMCP>  </PricingNodeInterval>  </PricingNode>  </RealTimeLMP>  </QueryResponse>  </env:Body>  </env:Envelope>    Error Handling   * + Invalid Transaction Point. Skip record.     - *No transaction point exists in the nMarket Database for the location name = <location>.*   + Invalid Type     - *Invalid date passed from the file <type> is not a valid value*   + Invalid Date     - *Invalid date passed from the file <day> is not a valid value.*   + Invalid Interval     - *Invalid hour passed from the file <interval> is not a valid value.*   + Invalid LMP     - *Invalid price passed from the file <LMP> is not a valid value.*   + Invalid MCC     - *Invalid price passed from the file <LMP> is not a valid value.*   + Invalid MLC     - *Invalid price passed from the file <LMP> is not a valid value.*   + Invalid Regulation MCP     - *Invalid price passed from the file <RegMCP> is not a valid value.*   + Invalid Regulation Mileage MCP     - *Invalid price passed from the file <RegMileageMCP> is not a valid value.*   + Invalid Spinning Reserve MCP     - *Invalid price passed from the file <SpinMCP> is not a valid value.*   + Invalid Supplemental Reserve MCP     - *Invalid price passed from the file <SuppMCP> is not a valid value.*       Testing Approach   |  |  | | --- | --- | | **ISO LOADS** | **(A)utomated / (M)anual** | | Connectivity - Download From Market | **N/A** | | FileListener | **N/A** | | Job Scheduler | **On hold** | | Load From File | **M** | | Verify price data appears correctly on View Prices screen. | **On hold** | | Load from File: Verify the correct Load from File process is recognized. | **On hold** | | Verify data is stored in the tables correctly. | **M** | | Verify that the start date and date interval values are determined correctly. | **M** | | Unique key validations for data integrity | **A** | | Date- monthly auctions, multi month auctions | **A** | | File with All the Valid Types that are supported (price types, Source) | **A** | | File with only one Type, should result in one record created | **A** | | XP\_Validate end-dated XP validation | **A** | | Validate the Auditpath and Load Summary window has correct information | **M** | | Error handling- Duplicate data validation for data integrity | **A** | | Error handling- Business Keys (required fields and unique keys) | **A** | | Error handling - Formatting | **A** | |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  | | |
|  |  |  |  |  |