Appalti&Contratti
e-procurement

Generatore Modelli

Funzioni speciali

Gennaio 2020

GRUPPO Maggioli

Sommario:

Informazioni sul documento

Versione

Versione	Data	Modifiche apportate
1.0	01/10/2008	Non applicabile in quanto questa è la prima versione del documento
2.0	16/05/2019	Aggiornamento complessivo del documento per restyling Aggiornamento comandi con nuova sintassi e rimossi quelli non più supportati
3.0	07/01/2020	Rimosso il capitolo "3 La "JOIN" implicita" perché comando deprecato

Scopo

Il presente documento descrive la funzionalità "Generatore modelli" che permette di comporre un documento a partire da un modello predefinito nel quale venga indicata un'apposita sintassi di programmazione.

Riferimenti ad altri documenti

Riferimenti
1. Generatore modelli: Funzioni base 2. Generatore modelli: Funzioni avanzate

Definizioni – acronimi – glossario

Termine – acronimo	Significato

1 Generatore modelli: istruzioni riservate a utenti esperti

Questo capitolo descrive le istruzioni più complesse del Generatore modelli riservate agli amministratori di sistema e programmatori esperti.

1.1 Prerequisiti

Per comprendere le istruzioni descritte in questa sezione del manuale è consigliabile conoscere i principali rudimenti sui database relazionali e il linguaggio SQL.

2 L'istruzione di collegamento e selezione [JOIN]

L'istruzione [JOIN] permette di stabilire relazioni tra dati (o record) di entità diverse, oppure più semplicemente di ricercare dati di argomenti collegati a quello che si sta trattando, impostando dei criteri di collegamento.

L'istruzione JOIN può quindi essere utilizzata per estrarre dati da entità diverse per estrarre una lista di occorrenze caratterizzate da una condizione particolare (specificata nel comando JOIN stesso), oppure quando queste non sono collegate dalle relazioni "padre-figlio" pre-configurate nell'applicazione, oppure quando si deve "navigare in senso inverso, cioè da "figlio" a "padre".
Un esempio può essere l'invio di una lettera ad un soggetto che riporti l'elenco delle ditte di cui egli è legale rappresentante servirà l'istruzione [JOIN].

ISTRUZIONE [JOIN]

Sintassi:	[JOIN]#MNEMONICO1#,#MNEMONICO2#,…=Param1, Param2, ... …… …… [ENDJOIN]
dove:	i riferimenti #MNEMONICO1#,#MNEMONICO2#,…indicano le chiavi sulle quali si assegnano i criteri di selezione; i criteri di selezione Param1, Param2, … definiscono i valori che devono essere assunti dalle chiavi nei record che si intendono selezionare; i criteri di selezione possono essere dei mnemonici, delle costanti, delle stringhe o dei totalizzatori.
Nota:	La selezione del comando [JOIN] rimane attiva fino a quando non si incontra il comando di fine [ENDJOIN].

L'istruzione [JOIN], quando è utilizzata per l'estrazione di una lista di dati, deve essere seguita dall'istruzione di ciclo [FOR] (vedi volume Funzioni Avanzate del Generatore Modelli, paragrafo relativo all'istruzione di ripetizione). Se si usa la [JOIN] per ricavare un solo record non è necessaria l'istruzione [FOR] ma, nei parametri della JOIN devono essere esplicitate le condizioni necessarie e sufficienti per estrarre il record desiderato.

Nella scrittura del codice, si consiglia inoltre di chiudere subito l'istruzione con [ENDJOIN] e poi proseguire con la scrittura dell'ulteriore codice all'interno dell'istruzione. Si eviterà così di incorrere nell'errore "sono rimaste [JOIN] aperte".

2.1 L'istruzione [JOIN] predefinita per collegare le entità padre-figlio

Il Generatore modelli consente di estrarre dati dall'entità principale (di partenza) del modello e dalle entità figlie (cioè quelle relative alle liste che derivano direttamente dalle maschere dell'entità principale) senza necessità di utilizzo di JOIN.
Nell'esempio delle imprese quindi i dati dei legali rappresentanti, dei direttori tecnici, delle categorie d'iscrizione ecc. sono elencabili direttamente con dei comandi di ciclo FOR.
Pur tuttavia se è necessario identificare un dato specifico di una particolare occorrenza di una entità figlia si può utilizzare ugualmente l'istruzione JOIN invece dei cicli con le condizioni interne ([IF]…).
Il comando di collegamento in questi casi assume la forma :

[JOIN]#CODfiglia#,#DATOfiglia#=#CODpadre#,"valore"

dove #CODpadre# è la chiave principale dell'entità padre, #CODfiglia# il corrispondente nella figlia, e #DATOfiglia# individua il campo della entità figlia utilizzato per individuare univocamente il record in base al "valore" assegnato.

Ad esempio per estrarre esclusivamente la data inizio incarico del legale rappresentante Rossi si può fare così:

[JOIN]#CODIMP2#,#NOMLEG#=#CODIMP#,"Rossi"
il sig. #NOMLEG# è legale rappresenta te dal #G_LEGINI#
[ENDJOIN]

lo stesso risultato si sarebbe ottenuto con la sequenza:
[FOR]II=1,99
@#NOMLEG#
[IF] %#NOMLEG#_EQ_Rossi%
il sig. #NOMLEG# è legale rappresenta te dal #G_LEGINI#
[ENDIF]
[NEXT]II

2.2 L'istruzione [JOIN] per navigare da figlio a padre

L'istruzione JOIN è indispensabile quando si opera con un modello che ha come entità principale una entità di dettaglio, come sono ad esempio le entità anagrafiche, e si vuole raggiungere qualche dato che si trova sulle entità predominanti del programma, o meglio quelle che normalmente sono denominate entità "padre" dell'entità di dettaglio perché a un record di queste corrispondono numerosi record delle entità figlie di dettaglio.

Il comando di collegamento in questi casi assume la forma :

[JOIN]#CODpadre#=#CODfiglia#

dove #CODpadre# è la chiave principale dell'entità padre e #CODfiglia# il corrispondente nella figlia.

Nell'esempio che segue un modello che si attiva da una scheda anagrafica di un tecnico di un'impresa richiama dati dell'impresa per cui costui lavora. Si noti che nel caso in esame un tecnico può essere collegato a più di una impresa, per cui esiste una tabella intermedia che fa da collegamento tra quella dei tecnici e quella delle imprese ed è "figlia" di entrambe. Il tutto è chiarito meglio nell'esempio.

Esempio:

```
@Questo modello eseguito dalla scheda anagrafica di un tecnico impresa
@Schema è GENE, Entità AN_TEC_IMP, chiave CODTIM
@#CODTIM#

Egregio #NOMTIM#
Via #INDTIM#, #NCITIM#
#LOCTIM# (#PROTIM.N#)

in qualità di legale rappresentante
@risalgo la gerarchia dall'anagrafica del tecnico Entità AN_TEC_IMP
```

```
@all'entità che collega il tecnico alle imprese LEG_RAP_IM che presenta
@le chiavi CODLEG e CODLEG
@Per selezionare con l'istruzione JOIN i record di interesse dovrò dire che
@CODLEG di LEG_RAP_IM è identico a CODLEG di AN_TEC_IMP
[JOIN]#CODIMP2#,#CODLEG#="*",#CODTIM#
[FOR]AA=1,99
@#CODIMP2#
[TOSTR]#STR1#=#G_LEGINI#
@risalgo nuovamente la gerarchia per giungere ai dati dell'impresa in AN_IMPRESE
@Per selezionare con l'istruzione JOIN i record di interesse dovrò dire che
@CODIMP di AN_IMPRESE è identico a CODIMP2 di LEG_RAP_IM
[JOIN]#CODIMP#=#CODIMP2#
[FOR]BB=1,1
[TOSTR]#STR2#=#NOMIMP#
[NEXT]BB
[ENDJOIN]
dal #STR1# della ditta #STR2#
[NEXT]AA
[ENDJOIN]

la informiamo che…
```

Il risultato del file composto sarà:

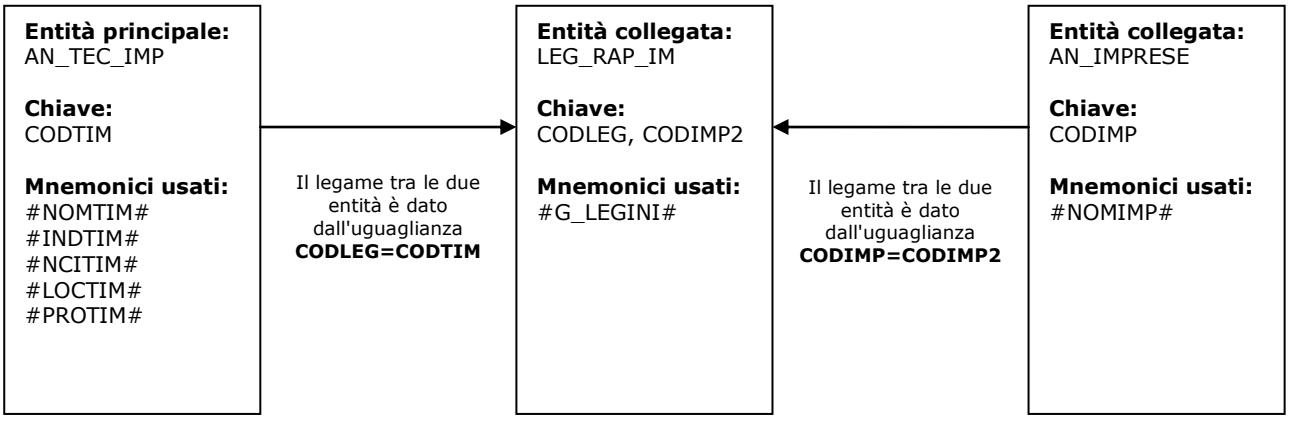
```
Egregio Mario Rossi
Via Via Londra, 14
Oderzo (TV)

in qualità di legale rappresentante
dal 25.05.2000 della ditta Fedele costruzioni snd
dal 01.01.2005 della ditta I.T. Impianti Tecnologici
dal 01.06.1997 della ditta EDILGAMMA

la informiamo che…
```

L'istruzione [JOIN]#CODIMP2#,#CODLEG#="*",#CODTIM# viene interpretata come: trova tutte le occorrenze di legali rappresentanti dell'entità LEG_RAP_IM, dove il codice dell'impresa iscritta #CODIMP2# è uno qualsiasi "*", mentre il codice #CODLEG# del legale rappresentante dell'entità LEG_RAP_IM è uguale al codice #CODTIM# dell'entità AN_TEC_IMP.
L'istruzione poteva essere scritta anche solo con [JOIN]#CODLEG#=#CODTIM#.

L'istruzione [JOIN]#CODIMP#=#CODIMP2# viene interpretata come: trova tutte le occorrenze di imprese (entità AN_IMPRESE) dove il codice impresa #CODIMP# è uguale al codice impresa #CODIMP2# a cui è iscritto il legale rappresentante (entità LEG_RAP_IM).



2.3 L'istruzione [JOIN] per effettuare una selezione di occorrenze

Il comando [JOIN] in combinazione con il comando di ciclo [FOR] è molto utilizzato per generare prospetti contenti dati di entità di dettaglio.
Il comandi assumono la sequenza:

[JOIN]#CODfiglia#=#CODpadre#
[FOR]IND=1,9999
@#CODfiglia#

#DATO1figlia# #DATO2figlia# #DATO3figlia# …
[NEXT]IND
[ENDJOIN]

Ad esempio si può stampare un prospetto i dati dei tecnici collegati ad una impresa: questo esempio è complicato dal fatto che l'entità figlia dell'impresa non contiene direttamente i dati del tecnico, e serve quindi una successiva JOIN che li estrae dall'entità anagrafica dei tecnici.

```
@Questo modello eseguito dalla scheda una impresa
@#CODIMP#

        TECNICI DELL'IMPRESA #NOMIMP#
```

N.PROG.	NOME	INDIRIZZO	TELEFONO	DATA INIZIO

```
[JOIN]#CODDTE#=#CODIMP#
[FOR]AA=1,99
@#CODDTE#
[JOIN]#COGTIM#=#CODDTE#
```

#AA#	#NOMTIM#	#INDTIM#, #NCITIM# - #LOCTIM#	#TELTIM#	#DINTIM#

```
[ENDJOIN]
[NEXT]AA
[ENDJOIN]
```

Il risultato del file composto sarà:

TECNICI DELL'IMPRESA Fedele costruzioni snc

N.PROG.	NOME	INDIRIZZO	TELEFONO	DATA INIZIO
1	Rossi ing. Mario	Via Londra, 14 - Oderzo	0422787788	12.12.2001
2	Bianchi arch. Gianni	Via Parigi, 4 - Conegliano	0438665533	11.11.2002

3 Eseguire un'istruzione select SQL per estrarre dati

Il Generatore possiede un comando speciale che permette di estrarre dati lanciando un'istruzione SELECT del linguaggio SQL direttamente nel database. L'istruzione SQL può essere scritta con qualsiasi complessità ma, per scriverla è necessario:

- conoscere i nomi dei campi e delle tabelle (o delle viste) realmente presenti nel database
- usare le esatte espressioni SQL richieste dal DBMS che si sta utilizzando (ci sono delle lievi differenze di linguaggio tra i diversi Oracle, SQLServer, Access: queste differenze sono limitate a particolari espressioni, ma se si utilizzano proprio quelle il modello perde la trasferibilità in altri sistemi)

Ricordiamo brevemente la struttura dell'istruzione SELECT nei linguaggi SQL:

SELECT campo1, campo2, …, campo n FROM tabella WHERE condizioni di ricerca

la SELECT utilizzata, una volta lanciata durante la composizione del modello, può produrre una struttura di dati di varia dimensione: in generale la si può immaginare come una tabella, formata da m righe e n colonne (i vari campo1, campo 2, …, campo n).

Il generatore mette a disposizione una serie di comandi per estrarre da questa struttura di dati i singoli campi della tabella, per mezzo di comandi di ripetizione (cicli $$) e di domini di formato.
Nel seguito sono descritti i comandi di esecuzione dell'istruzione e di estrazione dei dati dalla struttura prodotta.

3.1 Il comando EXECSQL

Il comando EXECSQL permette di eseguire una istruzione SQL (una select) nel data base in uso all'applicazione. L'istruzione SQL deve essere contenuta il una stringa STRnn del Generatore; la risposta del DBMS all'istruzione SQL (cioè il risultato) viene ricevuto dal Generatore in una struttura dati che è indirizzata da un'altra stringa STRmm: pertanto il comando è strutturato come una assegnazione [TOSTR] di una stringa sull'altra con un "dominio" di rappresentazione. La sintassi è:

[TOSTR]#STRmm#=#STRnn{X_EXECSQL}#

si considerano quindi due stringhe, STRnn e STRmm, una delle quali ospita l'istruzione SQL, l'altra il risultato (nn e mm sono valori numerici che individuano le stringhe).
Per immettere l'istruzione SQL nella prima stringa si utilizza ancora un comando [TOSTR], seguito eventualmente da comandi [CATSTR] per concatenare valori o altre stringhe.

Ad esempio, se si vogliono estrarre i direttori tecnici che sono anche legali rappresentanti delle relative imprese, si scrivono i comandi:

```
@#CODIMP#

[TOSTR]#STR1#="SELECT NOMTIM, NOMIMP FROM TEIM, IMPDTE, IMPLEG, IMPR WHERE
CODLEG=CODDTE AND CODDTE=CODTIM AND CODIMP=CODIMP3 "
[TOSTR]#STR#=#STR1{X_EXECSQL}#

……
```

se invece si vogliono gli stessi dati per la sola impresa il cui codice è contenuto nella stringa STR3, si scriverà:

```
@#CODIMP#

[TOSTR]#STR1#="SELECT NOMTIM, NOMIMP FROM TEIM, IMPDTE, IMPLEG, IMPR WHERE CODLEG=CODDTE AND CODDTE=CODTIM AND CODIMP='"
[CATSTR]#STR1#=#STR1#,#STR3#
[CATSTR]#STR1#=#STR1#,"'"
[TOSTR]#STR#=#STR1{X_EXECSQL}#

……
```

a questo punto l'istruzione SQL è eseguito: nei paragrafi successivi è descritto il modo di estrarre i risultati.

3.2 Quantità di righe estratte

I risultati del comando EXECSQL sono disponibili in una struttura dati identificata da una stringa STR. Poiché il comando può estrarre numerose occorrenze, ciascuna composta da diversi dati, la struttura può essere immaginata come una tabella composta da righe e colonne.

Per conoscere il numero di righe della tabella si può utilizzare un comando di operazione, con sintassi:

@ \#STRmm#\\++n\

dove : STRmm è la stringa che identifica la struttura che contiene il risultato,
n è il numero identificativo del totalizzatore TOTn che ospiterà il numero di occorrenze estratte

Nell'esempio del paragrafo precedente, in cui si è utilizzata la stringa senza numero, si potrà scrivere:

```
@#CODIMP#

[TOSTR]#STR1#="SELECT NOMTIM, NOMIMP FROM TEIM, IMPDTE, IMPLEG, IMPR WHERE CODLEG=CODDTE AND CODDTE=CODTIM AND CODIMP="
[CATSTR]#STR1#=#STR1#,#STR3#
[CATSTR]#STR1#=#STR1#,"'"
[TOSTR]#STR#=#STR1{X_EXECSQL}#
@    \#STR#\\++10\
Il numero di occorrenze trovate è #TOT10[N8]#
……
```

e il numero di occorrenze sarà ospitato dal totalizzatore TOT10:

```
Il numero di occorrenze trovate è       2
……
```

3.3 Il ciclo di estrazione delle righe

Ci proponiamo ora di estrarre dalla struttura dei risultati le occorrenze. Il numero di righe della struttura lo abbiamo individuato con il comando di cui al paragrafo precedente.
Il numero di dati per ciascuna occorrenza è legato all'istruzione SQL scritta e fatta eseguire, quindi è ben noto: è il numero di nomi di campi scritti dopo l'istruzione SELECT e prima della parola FROM.
Non resta che scrivere quindi il ciclo di ripetizioni che consente di stampare i dati estratti, sarà un ciclo privo di riferimenti ad una entità di data base, per cui applicheremo il comando speciale #_NOENT_# per dire al Generatore di non cercare entità; la sintassi sarà:

[FOR]IND=1,#TOTm#|#_NOENT_#

all'interno del gruppo di righe da ripetere metteremo i comandi di assegnazione a stringhe normali dei dati contenuti nell'occorrenza. La sintassi di questi comandi speciali di assegnazione sarà:

[TOSTR]#STR1#=#IND{X_SQLc1}#
[TOSTR]#STR2#=#IND{X_SQLc2}#
…
[TOSTR]#STRn#=#IND{X_SQLcn}#

dove : 1, 2, …. n sono i numeri d'ordine dei campi dati inseriti nella SELECT
IND è l'indice del ciclo ripetitivo

Il ciclo ripetitivo andrà a finire con un normale comando di termine :

[NEXT]IND

Nell'esempio del paragrafo precedente, in cui si era scritta l'istruzione SQL per estrarre il nome del direttore tecnico e il nome dell'impresa nei soli casi in cui il direttore tecnico era anche legale rappresentante, ci sono due dati per ciascuna occorrenza, per cui scriveremo:

```
@#CODIMP#

[TOSTR]#STR1#="SELECT NOMTIM, NOMIMP FROM TEIM, IMPDTE, IMPLEG, IMPR WHERE
CODLEG=CODDTE AND CODDTE=CODTIM AND CODIMP="
[CATSTR]#STR1#=#STR1#,#STR3#
[CATSTR]#STR1#=#STR1#,""""
[TOSTR]#STR#=#STR1{X_EXECSQL}#
@     \#STR#\\++10\
Il numero di occorrenze trovate è #TOT10[N8]#:
[FOR]JJ=1,#TOT10#|#_NOENT_#
[TOSTR]#STR1#=#JJ{X_SQLc1}#
[TOSTR]#STR2#=#JJ{X_SQLc2}#
- Impresa #STR2# : Direttore tecnico e legale rappresentante il sig. #STR1
[NEXT]JJ
……
```

il risultato sarà :

```
Il numero di occorrenze trovate è       2:
- Impresa Fedele Costruzioni s.n.c.: Direttore tecnico e legale rappresentante il sig. ing. Mario Rossi
- Impresa IMPRE.CO. s.r.l.: Direttore tecnico e legale rappresentante il sig. Alberto Russo
……
```

3.4 Gestire i risultati di molte SQLEXEC

Può capitare di dover lanciare più di una istruzione SQL di selezione e di dover poi gestirne i risultati nel resto del testo.
Per risolvere questo problema il programma mette a disposizione la cosiddetta "SQLEXEC nominata", ovvero un comando SQLEXEC dotato di nome, i quale consente di gestire contemporaneamente più di una struttura di memorizzazione dei dati.

Sintassi:

[TOSTR]#STRmm#=#STRnn{X_EXECSQL_nAAA}#

dove :
STRmm : stringa che indica la struttura di ricezione
STRnn : stringa che contiene l'istruzione SELECT
AAA : nome della struttura dei risultati (può essere un nome qualsiasi).

L'utilizzo è lo stesso della EXECSQL normale, visto nei paragrafi precedenti.
Anche la "conta" dei risultati è identica, basta richiamare in una operazione la stringa della struttura di ricezione:
@ \#STRmm#\\++n\

L'estrazione dei dati funziona in modo analogo: si definirà un ciclo senza entità e al suo interno di estrarranno i risultati utilizzando il dominio di formato X_SQLcn, stavolta corredato dal nome della struttura: _nAAA, dove AAA può essere un nome qualsiasi.

Sintassi:

[FOR]IND=1,#TOTm#|#_NOENT_#
[TOSTR]#STR1#=#IND{X_SQLc1_nAAA }#
[TOSTR]#STR2#=#IND{X_SQLc2_nAAA }#
…
[TOSTR]#STRn#=#IND{X_SQLcn_nAAA }#
… … …
[NEXT]IND

Si osservi che non è necessario che le stringhe di destinazione (STR1, STR2,..) abbiano lo stesso numero indicato nel dominio (X_SQLc1, X_SQLc2, …): il numero contenuto del dominio (c1, c2, … cn) indica l'ordine dei campi estratti dalla SELECT (campi 1, campo2, …, campo n) , ma nell'estrazione ogni campo può essere depositata su una stringa qualsiasi.