Talend Open Studio Composants

4.X

Guide de référence

Intentionally
Blank

Version 4.1_b

Convient pour l'utilisation de **Talend Open Studio** v4.1.x. Annule et remplace toute version antérieure du Guide de référence.

Copyleft

Cette documentation est mise à disposition selon les termes du Contrat Public Creative Commons (CPCC).

Pour plus d'informations concernant votre utilisation de cette documentation en accord avec le Contrat CPCC, consultez : http://creativecommons.org/licenses/by-nc-sa/2.0/

Talend Open Studio Composants Guide de référence i

Préface ..... **xxiii**
Objectif xxiii
Public visé xxiii
Conventions typographiques xxiii
Historique des versions ..... **xxiv**
Remarques et Support ..... **xxiv**

Composants Business ..... **xxv**

tAlfrescoOutput ..... **xxvi**
Propriétés du tAlfrescoOutput xxvi
Procédure d'installation xxviii
Dématérialisation, tAlfrescoOutput et Gestion de Contenu xxxi
Scénario : Créer des documents sur un serveur Alfresco xxxii
tBonitaDeploy ..... **xxxviii**
Propriétés du tBonitaDeploy xxxviii
Scénario associé xxxix
tBonitaInstantiateProcess ..... **xl**
tBonitaInstantiateProcess Properties xl
Scénario : Exécuter un processus Bonita via un Job Talend xli
tCentricCRMInput ..... **xlvi**
Propriétés du tCentricCRMInput xlvi
Scénario associé xlvi
tCentricCRMOutput ..... **xlvii**
Propriétés du tCentricCRMOutput xlvii
Scénario associé xlvii
tHL7Input ..... **xlviii**
Propriétés du tHL7Input xlviii
Scénario : Retrouver des informations concernant les patients et les événements à partir d'un fichier HL7 xlix
tHL7Output ..... **liii**
Propriétés du tHL7Output liii
Scénario associé liii
tMicrosoftCRMInput ..... **liv**
Propriétés du tMicrosoftCRMInput liv
Scénario : Ecrire des données dans une base de données Microsoft CRM et attribuer des conditions aux colonnes pour extraire des lignes spécifiques lv
tMicrosoftCRMOutput ..... **lxii**
Propriétés du tMicrosoftCRMOutput lxii
Scénario associé lxiii
tMSAXInput ..... **lxiv**
Propriétés du tMSAXInput lxiv
Scénario associé lxiv
tMSAXOutput ..... **lxv**
Propriétés du tMSAXOutput lxv
Scénario : Insérer des données dans une table définie sur le serveur Microsoft AX lxvi
Scénario 2 : Effacer des données d'une table précise sur le serveur MicrosoftAX. lxix
tOpenbravoERPInput ..... **lxxii**
Propriétés du tOpenbravoERPInput lxxii
Scénario associé lxxiii
tOpenbravoERPOutput ..... **lxxiv**
Propriétés du tOpenbravoERPOutput lxxiv
Scénario associé lxxiv
tSalesforceBulkExec ..... **lxxv**
Propriétés du tSalesforceBulkExec lxxv
Scénario associé lxxvi
tSalesforceConnection ..... **lxxvii**
Propriétés du tSalesforceConnection lxxvii
Scénario associé lxxvii
tSalesforceGetDeleted ..... **lxxviii**
Propriétés du tSalesforceGetDeleted lxxviii
Scénario : Récupérer les données supprimées du serveur Salesforce lxxix
tSalesforceGetServerTimestamp ..... **lxxxii**
Propriétés du tSalesforceGetServerTimestamp lxxxii
Scénario associé lxxxiii
tSalesforceGetUpdated ..... **lxxxiv**
Propriétés du tSalesforceGetUpdated lxxxiv
Scénario associé lxxxv
tSalesforceInput ..... **lxxxvi**
Propriétés du tSalesforceInput lxxxvi
Scénario : Utiliser des requêtes pour extraire des données d'une base de données Salesforce lxxxviii
tSalesforceOutput ..... **xcii**
Propriétés du tSalesforceOutput xcii
Scénario : Supprimer des données de l'objet Account xciv
tSalesforceOutputBulk ..... **xcvi**
Propriétés du tSalesforceOutputBulk xcvi
Scénario : Insérer des données de masse transformées dans votre Salesforce.com xcvi
tSalesforceOutputBulkExec ..... **cii**
Propriétés du tSalesforceOutputBulkExec cii
Scénario : Insérer des données de masse dans votre Salesforce.com ciii
tSAPCommit ..... **cvii**
Propriétés du tSAPCommit cvii
Scénario associé cvii
tSAPConnection ..... **cviii**
Propriétés du tSAPConnection cviii
Scénario associé cviii
tSAPInput ..... **cix**

Propriétés du tSAPInput cix
Scénario 1 : Récupérer les métadonnées d'un système SAP cxi
Scénario 2 : Lire les données des différents schémas de la fonction RFC_READ_TABLE cxvii
tSAPOutput cxxiii
Propriétés du tSAPOutput cxxiii
Scénarios associés cxxiv
tSAPRollback cxxv
Propriétés du tSAPRollback cxxv
Scénario associé cxxv
tSugarCRMInput cxxvi
Propriétés du tSugarCRMInput cxxvi
Scénario: Extraire les données d'un compte à partir de SugarCRM cxxvi
tSugarCRMOutput cxxix
Propriétés du tSugarCRMOutput cxxix
Scénario associé cxxix
tVtigerCRMInput cxxx
Propriétés du tVtigerCRMInput cxxx
Scénario associé cxxxi
tVtigerCRMOutput cxxxii
Propriétés du tVtigerCRMOutput cxxxii
Scénario associé cxxxiii

Composants Business Intelligence 1

tDB2SCD 2
Propriétés du tDB2SCD 2
Scénario associé 3
tDB2SCDELT 4
Propriétés du tDB2SCDELT 4
Scénario associé 6
tGreenplumSCD 7
Propriétés du tGreenplumSCD 7
Scénario associé 8
tInformixSCD 9
Propriétés du tInformixSCD 9
Scénario associé 10
tIngresSCD 11
Propriétés du tIngresSCD 11
Scénario associé 12
tMondrianInput 13
Propriétés du tMondrianInput 13
Scénario : Tables en jointure croisée 14
tMSSqlSCD 17
Propriétés du tMSSqlSCD 17
Scénario associé 18
tMysqlSCD 19
Propriétés du tMysqlSCD 19
Méthodologie de gestion du SCD 20
Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3 23
tMysqlSCDELT 32
Propriétés du tMysqlSCDELT 32
Scénario associé 34
tOracleSCD 35
Propriétés du tOracleSCD 35
Scénario associé 36
tOracleSCDELT 37
Propriétés du tOracleSCDELT 37
Scénario associé 39
tPaloCheckElements 40
Propriétés du tPaloCheckElements 40
Scénario associé 42
tPaloConnection 43
Propriétés du tPaloConnection 43
Scénario associé 43
tPaloCube 44
Propriétés du tPaloCube 44
Scénario : Créer un cube dans une base de données existante 46
tPaloCubeList 48
Propriétés du tPaloCubeList 48
Découvrir le schéma de sortie en lecture seule du tPaloCubeList 49
Scénario : Récupérer des informations détaillées d'un cube d'une base de données déterminée 50
tPaloDatabase 52
Propriétés du tPaloDatabase 52
Scénario : Créer une base de données 53
tPaloDatabaseList 55
Propriétés du tPaloDatabaseList 55
Découvrir le schéma de sortie en lecture seule du tPaloDatabaseList 56
Scénario : Récupérer des informations détaillées concernant les bases de données d'un serveur Palo donné 57
tPaloDimension 59
Propriétés du tPaloDimension 59
Scénario : Créer une dimension avec des éléments ... 63
tPaloDimensionList 68
Propriétés du tPaloDimensionList 68
Découvrir le schéma de sortie en lecture seule du tPaloDimensionList 70
Scénario: Récupérer des informations détaillées concernant les dimensions d'une base de données déterminée 70
tPaloInputMulti 72
Propriétés du tPaloInputMulti 72
Scénario : Récupérer des éléments de dimensions dans un cube donné 74
tPaloOutput 78
Propriétés du tPaloOutput 78
Scénario associé 79
tPaloOutputMulti 80

Propriétés du tPaloOutputMulti 80
Scénario 1 : Ecrire des données dans un cube donné 82
Scénario 2 : Refuser un flux de données d'entrée lorsque les éléments à écrire n'existent pas dans un cube donné 85
tPaloRule 89
Propriétés du tPaloRule 89
Scénario : Créer une règle dans un cube donné 90
tPaloRuleList 93
Propriétés du tPaloRuleList 93
Découvrir le schéma de sortie en lecture seule du tPaloRuleList 94
Scénario : Récupérer des informations détaillées concernant des règles d'un cube donné 95
tParAccelSCD 97
Propriétés du tParAccelSCD 97
Scénario associé 98
tPostgresPlusSCD 99
Propriétés du tPostgresPlusSCD 99
Scénario associé 100
tPostgresPlusSCDELT 101
Propriétés du tPostgresPlusSCDELT 101
Scénario associé 103
tPostgresqlSCD 104
Propriétés du tPostgresqlSCD 104
Scénario associé 105
tPostgresqlSCDELT 106
Propriétés du tPostgresqlSCDELT 106
Scénario associé 108
tSPSSInput 109
Propriétés du tSPSSInput 109
Scénario : Afficher le contenu d'un fichier SPSS 109
tSPSSOutput 113
Propriétés du tSPSSOutput 113
Scénario : Ecrire des données dans un fichier .sav 113
tSPSSProperties 116
Propriétés du tSPSSProperties 116
Scénario associé 116
tSPSSStructure 117
Propriétés du tSPSSStructure 117
Scénario associé 117
tSybaseSCD 119
Propriétés du tSybaseSCD 119
Scénario associé 120
tSybaseSCDELT 121
Propriétés du tSybaseSCDELT 121
Scénario associé 123

Composants Custom Code 125

tGroovy 126
Propriétés du tGroovy 126
Scénarios associés 126
tGroovyFile 127
Propriétés du tGroovyFile 127
Scénario : Appeler du code Groovy contenu dans un fichier 127
tJava 129
Propriétés du tJava 129
Scénario : Imprimer le contenu d'une variable 129
tJavaFlex 132
Propriétés du tJavaFlex 132
Scénario 1 : Générer un flux de données 133
Scénario 2 : Traiter des lignes de données avec le tJavaFlex 135
tJavaRow 139
Propriétés du tJavaRow 139
Scénario associé 139
tLibraryLoad 140
Propriétés du tLibraryLoad 140
Scénario : Vérifier le format d'une adresse e-mail 140

Composants Data Quality 143

tAddCRCRow 144
Propriétés du tAddCRCRow 144
Scénario : Ajouter une clé de substitution à un fichier 144
tExtractRegexFields 147
tFuzzyMatch 148
Propriétés du tFuzzyMatch 148
Scénario 1 : Distance de Levenshtein de 0 pour les prénoms 149
Scénario 2 : Distance de Levenshtein de 1 ou 2 pour les prénoms 151
Scénario 3 : Distance métaphonique pour les prénoms 152
tIntervalMatch 154
Scénario : Identifier le pays à partir de l'IP (Perl et Java) 155
tParseAddress 159
Propriétés du tParseAddress 159
Scénario associé 160
tParseName 161
Propriétés du tParseName 161
Scénario associé 162
tReplaceList 163
Propriétés du tReplaceList 163
Scénario : Remplacement à partir d'un fichier de référence 164
tSchemaComplianceCheck 168
Propriétés du tSchemaComplianceCheck 168

Scénario : Valider les dates en fonction d'un schéma (java) 169
tUniqRow 173
Propriétés du tUniqRow 173
Scénario : Dédoublonner des données 173
175

Composants Databases 177

tAccessBulkExec 178
Propriétés du tAccessBulkExec 178
Scénarios associés 180
tAccessInput 181
Propriétés du tAccessInput 181
Scénarios associés 182
tAccessOutput 183
Propriétés du tAccessOutput 183
Scénarios associés 185
tAccessOutputBulk 187
Propriétés du tAccessOutputBulk 187
Scénarios associés 188
tAccessOutputBulkExec 189
Propriétés du tAccessOutputBulkExec 189
Scénarios associés 190
tAccessRow 191
Propriétés du tAccessRow 191
Scénarios associés 192
tAS400Close 193
Propriétés du tAS400Close 193
Scénario associé 193
tAS400Commit 194
Propriétés du tAS400Commit 194
Scénario associé 194
tAS400Connection 195
Propriétés du tAS400Connection 195
Scénario associé 196
tAS400Input 197
Propriétés du tAS400Input 197
Scénarios associés 199
tAS400LastInsertId 200
Propriétés du tAS400LastInsertId 200
Scénario associé 200
tAS400Output 201
Propriétés du tAS400Output 201
Scénarios associés 205
tAS400Rollback 206
Propriétés du tAS400Rollback 206
Scénario associé 206
tAS400Row 207
Propriétés du tAS400Row 207
Scénarios associés 209
tCreateTable 210
Propriétés du tCreateTable 210
Scénario : Créer une nouvelle table dans une base de données MySQL 213
DB2BulkExec 215
Propriétés du tDB2BulkExec 215
Scénario associé 217
tDB2Close 218
Propriétés du tDB2Close 218
Scénario associé 218
tDB2Commit 219
Propriétés du tDB2Commit 219
Scénario associé 219
tDB2Connection 220
Propriétés du tDB2Connection 220
Scénarios associés 221
tDB2Input 222
Propriétés du tDB2Input 222
Scénarios associés 224
tDB2Output 225
Propriétés du tDB2Output 225
Scénarios associés 229
tDB2Rollback 230
Propriétés du tDB2Rollback 230
Scénario associé 230
tDB2Row 231
Propriétés du tDB2Row 231
Scénarios associés 233
tDB2SCD 234
tDB2SCDELT 235
tDB2SP 236
Propriétés du tDB2SP 236
Scénario associé 237
tDBInput 238
Propriétés du tDBInput 238
Scénario 1 : Afficher les données sélectionnées à partir d'une table 239
Scénario 2 : Utiliser la variable StoreSQLQuery .241
tDBOutput 243
Propriétés du tDBOutput 243
Scénario : Afficher la base de données en sortie 245
tDBSQLRow 248
Propriétés du tDBSQLRow 248
Scénario : Réinitialiser l'auto-incrémentation d'une base de données 249
tEXAInput 252
Propriétés du tEXAInput 252
Scénarios associés 254
tEXAOutput 255
Propriétés du tEXAOutput 255
Scénarios associés 257
tEXARow 258
Propriétés du tEXARow 258
Scénarios associés 259
tEXistConnection 260

Propriétés du tEXistConnection 260
Scénario associé 260
tEXistDelete 262
Propriétés du tEXistDelete 262
Scénario associé 263
tEXistGet 264
tEXistGet properties 264
Scénario associé 265
tEXistList 266
Propriétés du tEXistList 266
Scénario associé 267
tEXistPut 268
Propriétés du tEXistPut 268
Scénario associé 269
tEXistXQuery 270
Propriétés du tEXistXQuery 270
Scénario associé 271
tEXistXUpdate 272
Propriétés du tEXistXUpdate 272
Scénario associé 273
tFirebirdClose 274
Propriétés du tFirebirdClose 274
Scénario associé 274
tFirebirdCommit 275
Propriétés du tFirebirdCommit 275
Scénario associé 275
tFirebirdConnection 276
Propriétés du tFirebirdConnection 276
Scénario associé 277
tFirebirdInput 278
Propriétés du tFirebirdInput 278
Scénarios associés 279
tFirebirdOutput 280
Propriétés du tFirebirdOutput 280
Scénarios associés 282
tFirebirdRollback 283
Propriétés du tFirebirdRollback 283
Scénario associé 283
tFirebirdRow 284
Propriétés du tFirebirdRow 284
Scénarios associés 286
tGreenplumBulkExec 287
Propriétés du tGreenplumBulkExec 287
Scénarios associés 289
tGreenplumClose 290
Propriétés du tGreenplumClose 290
Scénario associé 290
tGreenplumCommit 291
Propriétés du tGreenplumCommit 291
Scénario associé 291
tGreenplumConnection 292
Propriétés du tGreenplumConnection 292
Scénarios associés 293
tGreenplumInput 294
Propriétés du tGreenplumInput 294
Scénarios associés 295
tGreenplumOutput 297
Propriétés du tGreenplumOutput 297
Scénarios associés 300
tGreenplumOutputBulk 301
Propriétés du tGreenplumOutputBulk 301
Scénarios associés 302
tGreenplumOutputBulkExec 303
Propriétés du tGreenplumOutputBulkExec 303
Scénarios associés 304
tGreenplumRollback 306
Propriétés du tGreenplumRollback 306
Scénario associé 306
tGreenplumRow 307
Propriétés du tGreenplumRow 307
Scénarios associés 309
tGreenplumSCD 310
tHiveClose 311
Propriétés du tHiveClose 311
Scénario associé 311
tHiveConnection 312
Propriétés du tHiveConnection 312
Scénarios associés 313
tHiveRow 314
Propriétés du tHiveRow 314
Scénarios associés 315
tHSQLDbInput 317
Propriétés du tHSQLDbInput 317
Scénarios associés 319
tHSQLDbOutput 321
Propriétés du tHSQLDbOutput 321
Scénarios associés 325
tHSQLDbRow 326
Propriétés du tHSQLDbRow 326
Scénarios associés 328
tInformixBulkExec 329
Propriétés du tInformixBulkExec 329
Scénario associé 332
tInformixClose 333
Propriétés du tInformixClose 333
Scénario associé 333
tInformixCommit 334
Propriétés du tInformixCommit 334
Scénario associé 334
tInformixConnection 335
Propriétés du tInformixConnection 335
Scénarios associés 336
tInformixInput 337
Propriétés du tInformixInput 337
Scénarios associés 339
tInformixOutput 340

Propriétés du tInformixOutput 340
Scénarios associés 343
tInformixOutputBulk 344
Propriétés du tInformixOutputBulk 344
Scénarios associés 345
tInformixOutputBulkExec 346
Propriétés du tInformixOutputBulkExec 346
Scénarios associés 349
tInformixRollback 350
Propriétés du tInformixRollback 350
Scénario associé 350
tInformixRow 351
Propriétés du tInformixRow 351
Scénarios associés 353
tInformixSCD 354
tInformixSP 355
Propriétés du tInformixSP 355
Scénarios associés 357
tIngresClose 358
Propriétés du tIngresClose 358
Scénario associé 358
tIngresCommit 359
Propriétés du tIngresCommit 359
Scénario associé 359
tIngresConnection 360
Propriétés du tIngresConnection 360
Scénario associé 360
tIngresInput 361
Propriétés du tIngresInput 361
Scénarios associés 362
tIngresOutput 364
Propriétés du tIngresOutput 364
Scénarios associés 366
tIngresRollback 367
Propriétés du tIngresRollback 367
Scénario associé 367
tIngresRow 368
Propriétés du tIngresRow 368
Scénarios associés 369
tIngresSCD 370
tInterbaseClose 371
Propriétés du tInterbaseClose 371
Scénario associé 371
tInterbaseCommit 372
Propriétés du tInterbaseCommit 372
Scénario associé 372
tInterbaseConnection 373
Propriétés du tInterbaseConnection 373
Scénarios associés 373
tInterbaseInput 374
Propriétés du tInterbaseInput 374
Scénarios associés 375
tInterbaseOutput 376
Propriétés du tInterbaseOutput 376
Scénarios associés 378
tInterbaseRollback 379
Propriétés du tInterbaseRollback 379
Scénario associé 379
tInterbaseRow 380
Propriétés du tInterbaseRow 380
Scénarios associés 382
tJavaDBInput 383
Propriétés du tJavaDBInput 383
Scénarios associés 384
tJavaDBOutput 385
Propriétés du tJavaDBOutput 385
Scénarios asoociés 387
tJavaDBRow 388
Propriétés du tJavaDBRow 388
Scénarios associés 389
tJDBCColumnList 390
Propriétés du tJDBCColumnList 390
Scénario associé 390
tJDBCClose 391
Propriétés du tJDBCClose 391
Scénario associé 391
tJDBCCommit 392
Propriétés du tJDBCCommit 392
Scénario associé 392
tJDBCConnection 393
Propriétés du tJDBCConnection 393
Scénario associé 394
tJDBCInput 395
Propriétés du tJDBCInput 395
Scénarios associés 397
tJDBCOutput 398
Propriétés du tJDBCOutput 398
Scénarios associés 401
tJDBCRollback 402
Propriétés du tJDBCRollback 402
Scénario associé 402
tJDBCRow 403
Propriétés du tJDBCRow 403
Scénarios associés 405
tJDBCSP 406
Propriétés du tJDBCSP 406
Scénarios associés 407
tJDBCTableList 408
Propriétés du tJDBCTableList 408
Scénario associé 408
tLDAPAttributesInput 409
Propriétés du tLDAPAttributesInput 409
Scénario associé 411
tLDAPInput 412
Propriétés du tLDAPInput 412
Scénario : Afficher le contenu filtré d’un annuaire

LDAP 414
tLDAPOutput 417
Propriétés du tLDAPOutput 417
Scénario : Editer des données dans un annuaire LDAP 419
tLDAPRenameEntry 422
Propriétés du tLDAPRenameEntry 422
Scénarios associés 423
tMaxDBInput 424
Propriétés du tMaxDBInput 424
Scénarios associés 425
tMaxDBOutput 426
Propriétés du tMaxDBOutput 426
Scénarios associés 428
tMaxDBRow 429
Propriétés du tMaxDBRow 429
Scénarios associés 430
tMSSqlBulkExec 431
Propriétés du tMSSqlBulkExec 431
Scénarios associés 433
tMSSqlClose 434
Propriétés du tMSSqlClose 434
Scénario associé 434
tMSSqlColumnList 435
Propriétés du tMSSqlColumnList 435
Scénario associé 435
tMSSqlCommit 436
Propriétés du tMSSqlCommit 436
Scénario associé 436
tMSSqlConnection 437
Propriétés du tMSSqlConnection 437
Scénario associé 438
tMSSqlInput 439
Propriétés du tMSSqlInput 439
Scénarios associés 441
tMSSqlLastInsertId 442
Propriétés du tMSSqlLastInsertId 442
Scénario associé 442
tMSSqlOutput 443
Propriétés du tMSSqlOutput 443
Scénarios associés 447
tMSSqlOutputBulk 448
Propriétés du tMSSqlOutputBulk 448
Scénarios associés 449
tMSSqlOutputBulkExec 450
Propriétés du tMSSqlOutputBulkExec 450
Scénarios associés 452
tMSSqlRollback 453
Propriétés du tMSSqlRollback 453
Scénario associé 453
tMSSqlRow 454
Propriétés du tMSSqlRow 454
Scénarios associés 456
tMSSqlSCD 457
tMSSqlSP 458
Propriétés du tMSSqlSP 458
Scénarios associés 460
tMSSqlTableList 461
Propriétés du tMSSqlTableList 461
Scénario associé 461
tMysqlBulkExec 462
Propriétés du tMysqlBulkExec 462
Scénarios associés 464
tMysqlClose 465
Propriétés du tMysqlClose 465
Scénario associé 465
tMysqlColumnList 466
Propriétés du tMysqlColumnList 466
Scénario : Itérer une table de base de données et lister le nom des colonnes de la table 466
tMysqlCommit 470
Propriétés du tMysqlCommit 470
Scénario associé 470
tMysqlConnection 471
Propriétés du tMysqlConnection 471
Scénario : Insérer des données dans des tables mère/fille 471
tMysqlInput 476
Propriétés du tMysqlInput 476
Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie 478
tMysqlLastInsertId 484
Propriétés du tMysqlLastInsertId 484
Scénario : Récupérer les ID des dernières entrées ajoutées 484
tMysqlOutput 489
Propriétés du tMysqlOutput 489
Scénario 1 : Ajouter une colonne et modifier les données 493
Scénario 2 : Effectuer des mises à jour dans une base de données 498
Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject 501
Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL 507
tMysqlOutputBulk 513
Propriétés du tMysqlOutputBulk 513
Scénario : Insérer des données transformées dans une base MySQL 514
tMysqlOutputBulkExec 518
Propriétés du tMysqlOutputBulkExec 518
Scénario : Insérer des données dans une base MySQL 520
tMysqlRollback 522
Propriétés du tMysqlRollback 522
Scénario : Annuler l'insertion de données dans des ta-

bles mère/fille 522
tMysqlRow 524
Propriétés du tMysqlRow 524
Scénario 1 : Supprimer et re-générer un index de table MySQL 526
Scénario 2 : Utiliser l'instance PreparedStatement pour faire une requête sur des données 527
tMysqlSCD 534
tMysqlSCDELT 535
tMysqlSP 536
Propriétés du tMysqlSP 536
Scénario : Trouver le libellé State à l'aide d'une procédure stockée 537
tMysqlTableList 541
Propriétés du tMysqlTableList 541
Scénario : Effectuer une opération d'itération sur des tables et en effacer le contenu grâce à un modèle SQL défini par l'utilisateur (SQL Template) 541
Scénario associé 545
tNetezzaBulkExec 546
Propriétés du tNetezzaBulkExec 546
Scénarios associés 548
tNetezzaClose 549
Propriétés du tNetezzaClose 549
Scénario associé 549
tNetezzaCommit 550
Propriétés du tNetezzaCommit 550
Scénario associé 550
tNetezzaConnection 551
Propriétés du tNetezzaConnection 551
Scénario associé 552
tNetezzaInput 553
Propriétés du tNetezzaInput 553
Scénarios associés 555
tNetezzaNzLoad 556
Propriétés du tNetezzaNzLoad 556
Charger des colonnes de type DATE, TIME et TIMESTAMP 562
Scénario associé 562
tNetezzaOutput 563
Propriétés du tNetezzaOutput 563
Scénarios associés 566
tNetezzaRollback 567
Propriétés du tNetezzaRollback 567
Scénario associé 567
tNetezzaRow 568
Propriétés du tNetezzaRow 568
Scénario associé 570
tOracleBulkExec 571
Propriétés du tOracleBulkExec 571
Scénario : Supprimer et insérer des données dans une base Oracle 574
tOracleClose 578
Propriétés du tOracleClose 578
Scénario associé 578
tOracleCommit 579
Propriétés du tOracleCommit 579
Scénario associé 579
tOracleConnection 580
Propriétés du tOracleConnection 580
Scénario associé 581
tOracleInput 582
Propriétés du tOracleInput 582
Scénarios associés 584
tOracleOutput 585
Propriétés du tOracleOutput 585
Scénarios associés 589
tOracleOutputBulk 591
Propriétés du tOracleOutputBulk 591
Scénarios associés 592
tOracleOutputBulkExec 593
Propriétés du tOracleOutputBulkExec 593
Scénarios associés 596
tOracleRollback 597
Propriétés du tOracleRollback 597
Scénario associé 597
tOracleRow 598
Propriétés du tOracleRow 598
Scénario associé 600
tOracleSCD 601
tOracleSCDELT 602
tOracleSP 603
Propriétés du tOracleSP 603
Scénario : Vérifier le format de numéros à l'aide d'une procédure stockée 605
tOracleTableList 610
Propriétés du tOracleTableList 610
Scénario associé 610
tParAccelBulkExec 611
Propriétés du tParAccelBulkExec 611
Scénarios associés 613
tParAccelClose 614
Propriétés du tParAccelClose 614
Scénario associé 614
tParAccelCommit 615
Propriétés du tParAccelCommit 615
Scénario associé 615
tParAccelConnection 616
Propriétés du tParAccelConnection 616
Scénario associé 617
tParAccelInput 618
Propriétés du tParAccelInput 618
Scénarios associés 620
tParAccelOutput 621
Propriétés du tParAccelOutput 621
Scénarios associés 624

tParAccelOutputBulk 625
Propriétés du tParAccelOutputBulk 625
Scénarios associés 626
tParAccelOutputBulkExec 627
Propriétés du tParAccelOutputBulkExec 627
Scénarios associés 628
tParAccelRollback 630
Propriétés du tParAccelRollback 630
Scénario associé 630
tParAccelRow 631
Propriétés du tParAccelRow 631
Scénarios associés 633
tParAccelSCD 634
tParseRecordSet 635
Propriétés du tParseRecordSet 635
Scénario associé 635
tPostgresPlusBulkExec 636
Propriétés du tPostgresPlusBulkExec 636
Scénario associé 638
tPostgresPlusClose 639
Propriétés du tPostgresPlusClose 639
Scénario associé 639
tPostgresPlusCommit 640
Propriétés du tPostgresPlusCommit 640
Scénario associé 640
tPostgresPlusConnection 641
Propriétés du tPostgresPlusConnection 641
Scénario associé 642
tPostgresPlusInput 643
Propriétés du tPostgresPlusInput 643
Scénarios associés 645
tPostgresPlusOutput 646
Propriétés du tPostgresPlusOutput 646
Scénarios associés 649
tPostgresPlusOutputBulk 651
Propriétés du tPostgresPlusOutputBulk 651
Scénarios associés 652
tPostgresPlusOutputBulkExec 653
Propriétés du tPostgresplusOutputBulkExec 653
Scénarios associés 654
tPostgresPlusRollback 655
Propriétés du tPostgresPlusRollback 655
Scénario associé 655
tPostgresPlusRow 656
Propriétés du tPostgresPlusRow 656
Scénarios associés 658
tPostgresPlusSCD 659
tPostgresPlusSCDELT 660
tPostgresqlBulkExec 661
Propriétés du tPostgresqlBulkExec 661
Scénarios associés 663
tPostgresqlClose 664
Propriétés du tPostgresqlClose 664
Scénario associé 664
tPostgresqlCommit 665
Propriétés du tPostgresqlCommit 665
Scénario associé 665
tPostgresqlConnection 666
Propriétés du tPostgresqlConnection 666
Scénario associé 667
tPostgresqlInput 668
Propriétés du tPostgresqlInput 668
Scénarios associés 670
tPostgresqlOutput 671
Propriétés du tPostgresqlOutput 671
Scénarios associés 675
tPostgresqlOutputBulk 676
Propriétés du tPostgresqlOutputBulk 676
Scénarios associés 677
tPostgresqlOutputBulkExec 678
Propriétés du tPostgresqlOutputBulkExec 678
Scénarios associés 680
tPostgresqlRollback 681
Propriétés du tPostgresqlRollback 681
Scénario associé 681
tPostgresqlRow 682
Propriétés du tPostgresqlRow 682
Scénarios associés 684
tPostgresqlSCD 685
tPostgresqlSCDELT 686
tSASInput 687
Propriétés du tSASInput 687
Scénarios associés 688
tSASOutput 690
Propriétés du tSASOutput 690
Scénarios associés 692
tSQLiteClose 693
Propriétés du tSQLiteClose 693
Scénario associé 693
tSQLiteCommit 694
Propriétés du tSQLiteCommit 694
Scénario associé 694
tSQLiteConnection 695
Propriétés du tSQLiteConnection 695
Scénarios associés 696
tSQLiteInput 697
Propriétés du tSQLiteInput 697
Scénario : Filtrer des données SQlite 699
tSQLiteOutput 701
Propriétés du tSQLiteOutput 701
Scénario associé 704
tSQLiteRollback 705
Propriétés du tSQLiteRollback 705
Scénario associé 705
tSQLiteRow 706
Propriétés du tSQLiteRow 706

Scénario : Mettre à jour des lignes SQLite 708
tSybaseBulkExec 710
Propriétés du tSybaseBulkExec 710
Scénarios associés 712
tSybaseClose 713
Propriétés du tSybaseClose 713
Scénario associé 713
tSybaseCommit 714
Propriétés du tSybaseCommit 714
Scénario associé 714
tSybaseConnection 715
Propriétés du tSybaseConnection 715
Scénarios associés 715
tSybaseInput 716
Propriétés du tSybaseInput 716
Scénario associé 718
tSybaseIQBulkExec 719
Propriétés du tSybaseIQBulkExec 719
Scénario associé 721
tSybaseIQOutputBulkExec 722
Propriétés du tSybaseIQOutputBulkExec 722
Scénarios associés 724
tSybaseOutput 725
tSybaseOutput Properties 725
Scénario associé 729
tSybaseOutputBulk 730
Propriétés du tSybaseOutputBulk 730
Scénario associé 731
tSybaseOutputBulkExec 732
Propriétés du tSybaseOutputBulkExec 732
Scénarios associés 734
tSybaseRollback 735
Propriétés du tSybaseRollback 735
Scénario associé 735
tSybaseRow 736
Propriétés du tSybaseRow 736
Scénario associé 738
tSybaseSCD 739
tSybaseSCDELT 740
tSybaseSP 741
Propriétés du tSybaseSP 741
Scénario associé 743
tTeradataClose 744
Propriétés du tTeradataClose 744
Scénario associé 744
tTeradataCommit 745
Propriétés du tTeradataCommit 745
Scénario associé 745
tTeradataConnection 746
Propriétés du tTeradataConnection 746
Scénario associé 747
tTeradataFastExport 748
Propriétés du tTeradataFastExport 748
Scénario associé 749
tTeradataFastLoad 750
Propriétés du tTeradataFastLoad 750
Scénario associé 751
tTeradataFastLoadUtility 752
Propriétés du tTeradataFastLoadUtility 752
Scénario associé 753
tTeradataInput 754
Propriétés du tTeradataInput 754
Scénario associé 756
tTeradataMultiLoad 757
Propriétés du tTeradataMultiLoad 757
Scénario associé 759
tTeradataOutput 760
Propriétés du tTeradataOutput 760
Scénario associé 764
tTeradataRollback 765
Propriétés du tTeradataRollback 765
Scénario associé 765
tTeradataRow 766
Propriétés du tTeradataRow 766
Scénario associé 768
tTeradataTPump 769
Propriétés du tTeradataTPump 769
Scénario : Insérer des données dans une table d'une base de données Teradata 771
tVectorWiseCommit 775
Propriétés du tVectorWiseCommit 775
Scénario associé 775
tVectorWiseConnection 776
Propriétés du tVectorWiseConnection 776
Scénario associé 777
tVectorWiseInput 778
Propriétés du tVectorWiseInput 778
Scénario associé 780
tVectorWiseOutput 781
Propriétés du tVectorWiseOutput 781
Scénario associé 784
tVectorWiseRollback 785
Propriétés du tVectorWiseRollback 785
Scénario associé 785
tVectorWiseRow 786
Propriétés du tVectorWiseRow 786
Scénario associé 788
tVerticaBulkExec 789
Propriétés du tVerticaBulkExec 789
Scénarios associés 791
tVerticaClose 792
Propriétés du tVerticaClose 792
Scénario associé 792
tVerticaCommit 793
Propriétés du tVerticaCommit 793
Scénario associé 793

tVerticaConnection **794**
Propriétés du tVerticaConnection 794
Scénario associé 795
tVerticaInput **796**
Propriétés du tVerticaInput 796
Scénarios associés 798
tVerticaOutput **799**
Propriétés du tVerticaOutput 799
Scénarios associés 802
tVerticaOutputBulk **803**
Propriétés du tVerticaOutputBulk 803
Scénarios associés 804
tVerticaOutputBulkExec **805**
Propriétés du tVerticaOutputBulkExec 805
Scénarios associés 806
tVerticaRollback **807**
Propriétés du tVerticaRollback 807
Scénario associé 807
tVerticaRow **808**
Propriétés du tVerticaRow 808
Scénario associé 810

Composants ELT 811

tELTAggregate **812**
Propriétés du tELTAggregate 812
Scénario : Filtrer et agréger les colonnes d'une table directement dans le SGBD 814
tELTCommit **819**
Propriétés du tELTCommit 819
Scénario associé 820
tELTFilterColumns **821**
Propriétés du tELTFilterColumns 821
Scénario associé 822
tELTFilterRows **823**
Propriétés du tELTFilterRows 823
Scénario associé 824
tELTJDBCInput **825**
Propriétés de tELTJDBCInput 825
Scénarios associés 825
tELTJDBCMap **827**
Propriétés de tELTJDBCMap 827
Scénarios associés 828
tELTJDBCOutput **829**
Propriétés de tELTJDBCOutput 829
Scénarios associés 830
tELTMSSqlInput **831**
Propriétés de tELTMSSqlInput 831
Scénarios associés 831
tELTMSSqlMap **833**
Propriétés de tELTMSSqlMap 833
Scénarios associés 834
tELTMSSqlOutput **835**
Propriétés de tELTMSSqlOutput 835
Scénarios associés 836
tELTMysqlInput **837**
Propriétés de tELTMysqlInput 837
Scénarios associés 837
tELTMysqlMap **838**
Propriétés de tELTMysqlMap 838
Connecter les composants ELT 839
Tables de jointure et de correspondance 839
Ajouter des clauses Where 839
Générer la requête SQL 840
Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre 840
Scénario 2 : ELT utilisant une table Alias 843
tELTMysqlOutput **847**
Propriétés de tELTMysqlOutput 847
Scénarios associés 848
tELTOracleInput **849**
Propriétés de tELTOracleInput 849
Scénario associé 850
tELTOracleMap **851**
Propriétés de tELTOracleMap 851
Connecter les composants ELT 853
Tables de jointure et de correspondance 853
Ajouter des clauses Where 853
Générer la requête SQL 853
Scénario : Mettre à jour les lignes d'une base Oracle 853
tELTOracleOutput **856**
Propriétés de tELTOracleOutput 856
Scénario : Utiliser la fonction MERGE d'Oracle pour mettre à jour et insérer simultanément des données ..857
tELTPostgresqlInput **864**
Propriétés de tELTPostgresqlInput 864
Scénarios associés 864
tELTPostgresqlMap **866**
Propriétés de tELTPostgresqlMap 866
Scénarios associés 867
tELTPostgresqlOutput **868**
Propriétés de tELTPostgresqlOutput 868
Scénarios associés 869
tELTRollback **870**
Propriétés du tELTRollback 870
Scénario associé 871
tELTSybaseInput **872**
Propriétés de tELTSybaseInput 872
Scénarios associés 872
tELTSybaseMap **874**
Propriétés de tELTSybaseMap 874
Scénarios associés 875
tELTSybaseOutput **876**
Propriétés de tELTSybaseOutput 876
Scénarios associés 877

tELTTeradataInput **878**
Propriétés de tELTTeradataInput 878
Scénarios associés 878
tELTTeradataMap **879**
Propriétés de tELTTeradataMap 879
Connecter les composants ELT 880
Tables de jointure et de correspondance 880
Ajouter des clauses Where 880
Générer la requête SQL 880
Scénarios associés 880
tELTTeradataOutput **881**
Propriétés de tELTTeradataOutput 881
Scénarios associés 882

Composants File 883

tAdvancedFileOutputXML **884**
tApacheLogInput **885**
Propriétés du tApacheLogInput 885
Scénario : Lire un fichier de log d'accès Apache 886
tCreateTemporaryFile **888**
Propriétés du tCreateTemporaryFile 888
Scénario : Créer un fichier temporaire et y écrire des données 889
tFileArchive **893**
Propriétés du tFileArchive 893
Scénario : Zipper des fichiers à l'aide d'un tFileArchive 894
tFileCompare **896**
Propriétés du tFileCompare 896
Scénario : Comparer des fichiers dézippés 897
tFileCopy **900**
Propriétés du tFileCopy 900
Scénario : Récupérer un fichier de la corbeille ... 901
tFileDelete **903**
Propriétés du tFileDelete 903
Scénario : Supprimer des fichiers 903
tFileExist **906**
Propriétés du tFileExist 906
Scénario : Vérifier l'existence d'un fichier et le créer s'il n'existe pas 907
tFileInputARFF **911**
Propriétés du tFileInputARFF 911
Scénario : Afficher le contenu d'un fichier ARFF 912
tFileInputDelimited **915**
Propriétés du tFileInputDelimited 915
Scénario 1 : Afficher le contenu d'un fichier délimité 917
Scénario 2 : Lire les données d'un fichier distant en mode stream 919
tFileInputEBCDIC **923**
Propriétés du tFileInputEBCDIC 923
Scénario : Extraire les données d'un fichier EBCDIC et alimenter une base de données 923
tFileInputExcel **929**
Propriétés du tFileInputExcel 929
Scénario associé 931
tFileInputFullRow **932**
Propriétés du tFileInputFullRow 932
Scénario : Lire des lignes complètes dans un fichier délimité 932
tFileInputJSON **935**
Propriétés du tFileInputJSON 935
Scénario : Extraire les données des champs d'un fichier au format JSON 936
tFileInputLDIF **938**
Propriétés du tFileInputLDIF 938
Scénario associé 939
tFileInputMail **940**
Propriétés du tFileInputMail 940
Scénario : Extraire des champs clés d'un email ...941
tFileInputMSDelimited **943**
Propriétés du tFileInputMSDelimited 943
L'éditeur de schémas Multi Schema Editor 943
Scénario : Lecture d'un fichier délimité multi structure 945
tFileInputMSPositional **950**
Propriétés du tFileInputMSPositional 950
Scénario associé 951
tFileInputMSXML **952**
Propriétés du tFileInputMSXML 952
Scénario : Lecture d'un fichier XML multi structuré 953
tFileInputPositional **956**
Propriétés du tFileInputPositional 956
Scénario : Transformer un fichier positionnel en XML 958
tFileInputProperties **961**
Propriétés du tFileInputProperties 961
Scénario : Lire et mapper la clé et les valeurs de fichiers properties et alimenter un glossaire 961
tFileInputRegex **965**
Propriétés du tFileInputRegex 965
Scénario : Transformer en fichier Regex en Positional 966
tFileInputXML **969**
tFileList **970**
Propriétés du tFileList 970
Scénario : Itération sur un répertoire 972
tFileOutputARFF **976**
Propriétés du tFileOutputARFF 976
Scénario associé 977
tFileOutputDelimited **978**
Propriétés du tFileOutputDelimited 978

Scénario : Ecrire des données dans un fichier délimité 980
tFileOutputEBCDIC 985
Propriétés du tFileOutputEBCDIC 985
Scénario : Créer un fichier EBCDIC à partir de deux fichiers délimités 985
tFileOutputExcel 988
Propriétés du tFileOutputExcel 988
Scénario associé 989
tFileOutputJSON 990
Propiétés du tFileOutputJSON 990
Scénario : Ecrire un fichier JSON structuré 990
tFileOutputLDIF 994
Propriétés du tFileOutputLDIF 994
Scénario : Ecrire des données dans un fichier LDIF . 995
tFileOutputMSDelimited 998
Propriétés du tFileOutputMSDelimited 998
Scénario associé 999
tFileOutputMSPositional 1000
Propriétés du tFileOutputMSPositional 1000
Scénario associé 1000
tFileOutputMSXML 1001
Propriétés du tFileOutputMSXML 1001
Définir un arbre XML Multischéma 1001
Mapping de données XML à partir de sources multischéma 1004
Définir le statut du noeud 1005
Scénario associé 1006
tFileOutputPositional 1007
Propriétés du tFileOutputPositional 1007
Scénario associé : 1008
tFileOutputProperties 1009
Propriétés du tFileOutputProperties 1009
Scénario associé 1009
tFileOutputXML 1010
tFileProperties 1011
Propriétés du tFileProperties 1011
Scénario : Afficher les propriétés d'un fichier traité . 1012
tFileRowCount 1014
Propriétés du tFileRowCount 1014
Scénario associé : 1015
tFileTouch 1016
Propriétés du tFileTouch 1016
Scénario associé : 1016
tFileUnarchive 1017
Propriétés du tFileUnarchive 1017
Scénario associé 1018
tGPGDecrypt 1019
Propriétés du tGPGDecrypt 1019
Scénario : Décrypter un fichier crypté GnuPG et afficher son contenu 1019
tPivotToColumnsDelimited 1023
Propriétés du tPivotToColumnsDelimited 1023
Scénario : Utiliser une colonne pivot pour agréger des données 1023

Composants Internet 1027

tFileFetch 1028
Propriétés du tFileFetch 1028
Scénario 1 : Récupérer des données à partir d'un protocole HTTP 1030
Scénario 2 : Réutiliser un cookie stocké pour récupérerdes fichiers via un protocole HTTP 1031
Scénario associé 1034
tFileInputJSON 1035
tFTPConnection 1036
Propriétés du tFTPConnection 1036
Scénarios associés 1037
tFTPDelete 1038
Propriétés du tFTPDelete 1038
Scénarios associés 1039
tFTPFileExist 1040
Propriétés du tFTPFileExist 1040
Scénarios associés 1041
tFTPFileList 1042
Propriétés du tFTPFileList 1042
Scénario : Itération sur un répertoire distant 1043
tFTPFileProperties 1047
Propriétés du tFTPFileProperties 1047
Scénario associé 1048
tFTPGet 1049
Propriétés du tFTPGet 1049
Scénario associé 1051
tFTPPut 1052
Propriétés du tFTPPut 1052
Scénario : Copier des fichiers sur un serveur FTP distant 1053
tFTPRename 1056
Propriétés du tFTPRename 1056
Scénario associé 1057
tFTPTruncate 1058
Propriétés du tFTPTruncate 1058
Scénario associé 1059
tJMSInput 1060
Propriétés du tJMSInput 1060
Scénario associé : 1061
tJMSOutput 1062
Propriétés du tJMSOutput 1062
Scénario associé : 1062
tMicrosoftMQInput 1063
Propriétés du tMicrosoftMQInput 1063
Scénario : Ecrire et récupérer des messages dans une file d'attente de message Microsoft 1064

tMicrosoftMQOutput 1067
Propriétés du tMicrosoftMQOutput 1067
Scénario associé 1067
tMomInput 1068
Propriétés du tMomInput 1068
Scénario : Communication asynchrone via un serveur MOM 1069
tMomMessageIdList 1072
Propriétés du tMomMessageIdList 1072
Scénario associé 1072
tMomOutput 1073
Propriétés du tMomOutput 1073
Scénario associé 1073
tPOP 1074
Propriétés du tPOP 1074
Scénario : Récupérer une sélection d'e-mails à partir d'un compte de messagerie 1075
tRSSInput 1078
Propriétés du tRSSInput 1078
Scénario : Récupérer les billets fréquemment mis à jour sur un blog 1078
tRSSOutput 1081
Propriétés du tRSSOutput 1081
Scénario 1 : Créer un flux RSS et stocker les fichiers sur un serveur FTP 1082
Scénario 2 : Créer un flux RSS contenant des méta-données 1086
Scénario 3 : Créer un fichier XML de flux ATOM ... 1089
tSCPClose 1093
Propriétés du tSCPClose 1093
Scénario associé 1093
tSCPConnection 1094
Propriétés du tSCPConnection 1094
Scénarios associés 1094
tSCPDelete 1095
Propriétés du tSCPDelete 1095
Scénario associé 1095
tSCPFileExists 1096
Propriétés du tSCPFileExists 1096
Scénario associé 1096
tSCPFileList 1097
Propriétés du tSCPFileList 1097
Scénario associé 1097
tSCPGet 1098
Propriétés du tSCPGet 1098
Scénario : Récupérer des fichiers d'un serveur SCP distant 1098
tSCPPut 1100
Propriétés du tSCPPut 1100
Scénario associé 1100
tSCPRename 1101
Propriétés du tSCPRename 1101
Scénario associé 1101
tSCPTruncate 1102
Propriétés du tSCPRename 1102
Scénario associé 1102
tSendMail 1103
Propriétés du tSendMail 1103
Scénario : Envoyer un e-mail lors d'une erreur . 1104
tSetKeystore 1108
Propriétés du tSetKeystore 1108
Scénario : Extraire des informations client d'un fichier WSDL privé 1109
tSocketInput 1114
Propriétés du tSocketInput 1114
Scénario : Transférer des données vers un port d'écoute (Java) 1116
tSocketOutput 1119
Propriétés du tSocketOutput 1119
Scénario associé 1120
tSOAP 1121
Propriétés du tSOAP 1121
Scénario : Extraire des informations météo en utilisant un Service Web 1123
tWebServiceInput 1125
Propriétés du tWebServiceInput 1125
Scénario 1 : Extraire des images à l'aide d'un Web-service 1127
Scénario 2 : Lire des données publiées dans un service Web, en utilisant les fonctions avancées du tWebServiceInput (Java) 1128
tXMLRPCInput 1134
Propriétés du tXMLRPCInput 1134
Scénario : Chercher le nom d'un Etat via une méthode XMLRPC 1134

Composants Logs & Errors 1137

tAssert 1138
Propriétés du tAssert 1138
1138
Scénario : Paramétrer une condition assertive pour l'exécution d'un Job 1138
tAssertCatcher 1144
Propriétés du tAssertCatcher 1144
Scénario associé 1145
tChronometerStart 1146
Propriétés du tChronometerStart 1146
Scénario associé 1146
tChronometerStop 1147
Propriétés du tChronometerStop 1147
Scénario : Mesurer la durée de traitement d'un sous-Job ou d'une partie d'un sous-Job 1147
tDie 1152
Propriétés du tDie 1152

Scénarios associés 1152
tFlowMeter 1153
Propriétés du tFlowMeter 1153
Scénario associé 1153
tFlowMeterCatcher 1154
Propriétés du tFlowMeterCatcher 1154
Scénario : Mesurer le flux d'un Job 1155
tLogCatcher 1159
Propriétés du tLogCatcher 1159
Scénario 1 : Avertissement & log lors de la création d'entrées 1159
Scénario 2 : Monitorer & arrêter un Job 1161
tLogRow 1163
Propriétés du tLogRow 1163
Scénarios associés 1163
tStatCatcher 1164
Propriétés du tStatCatcher 1164
Scénario : Afficher les statistiques du Job 1164
tWarn 1167
Propriétés du tWarn 1167
Scénarios associés 1167

Composants Misc 1169

tAddLocationFromIP 1170
Propriétés du tAddLocationFromIP 1170
Scénario : Identifier la localisation géographique d'une adresse IP 1171
tBufferInput 1174
Propriétés du tBufferInput 1174
Scénario : Récupérer les données mises en tampon (Java) 1174
tBufferOutput 1177
Propriétés du tBufferOutput 1177
Scénario 1 : Mettre des données en tampon (Java) ... 1177
Scénario 2 : Mettre les données de sortie en mémoire tampon du serveur d'application Web 1180
Scénario 3 : Appeler un Job contenant des variables de contexte à partir de votre navigateur Web 1184
Scénario 4 : Appeler un Job exporté en tant que service Web dans un autre Job 1185
tContextDump 1188
Propriétés du tContextDump 1188
Scénario associé 1188
tContextLoad 1189
Propriétés du tContextLoad 1189
Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL 1190
tFixedFlowInput 1193
Propriétés du tFixedFlowInput 1193
Scénarios associés 1193
tMemorizeRows 1195
Propriétés du tMemorizeRows 1195
Scénario : Compter le nombre d'occurrences de différents âges 1196
tMsgBox 1202
Propriétés du tMsgBox 1202
Scénario : Test de type 'Hello world!' 1202
tRowGenerator 1204
Propriétés du tRowGenerator 1204
Définition du schéma 1204
Définition d'une fonction 1205
Scénario : Génération aléatoire de fichiers de test 1206

Composants Orchestration 1209

tFileList 1210
tFlowToIterate 1211
Propriétés du tFlowToIterate 1211
Scénario : Transformer un flux de données en liste ... 1211
tForeach 1216
Propriétés du tForeach 1216
Scénario : Faire une boucle sur une liste et récupérer les valeurs 1216
tInfiniteLoop 1219
Propriétés du tInfiniteLoop 1219
Scénario associé 1219
tIterateToFlow 1220
Propriétés du tIterateToFlow 1220
Scénario : Transformer une liste de fichiers en flux de données 1221
tLoop 1223
Propriétés du tLoop 1223
Scénario : Exécuter un Job à partir d'une boucle 1224
tPostjob 1226
Propriétés du tPostjob 1226
Scénario associé 1226
tPrejob 1227
Propriétés du tPrejob 1227
Scénario associé 1227
tReplicate 1228
Propriétés du tReplicate 1228
Scénario associé 1228
tRunJob 1229
tSleep 1230
Propriétés du tSleep 1230
Scénario associé 1230
tUnite 1231
Propriétés du tUnite 1231
Scénario : Itération sur des fichiers et fusion de contenu 1232
tWaitForFile 1235

Propriétés du tWaitForFile 1235
Scénario : Attendre la suppression d'un fichier 1237
tWaitForSocket 1240
Propriétés du tWaitForSocket 1240
Scénario associé 1241
tWaitForSqlData 1242
Propriété du tWaitForSqlData 1242
Scénario : Attendre l'ajout de lignes dans une table 1243

Composants Processing 1247

tAggregateRow 1248
Propriétés du tAggregateRow 1248
Scénario : Agréger des valeurs et trier des données .. 1250
tAggregateSortedRow 1254
Propriétés du tAggregateSortedRow 1254
Scénario associé 1255
tConvertType 1256
Propriétés du tConvertType 1256
Scénario : Convertir des types de données Java 1257
tDenormalize 1261
Propriétés du tDenormalize 1261
Scénario 1 : Dénormaliser une colonne 1262
Scénario 2 : Dénormaliser plusieurs colonnes .. 1263
tDenormalizeSortedRow 1266
Propriétés du tDenormalizeSortedRow 1266
Scénario : Regrouper des lignes triées 1266
tEmptyToNull 1270
Propriétés du tEmptyToNull 1270
Scénario : Remplacer des champs vides par des champs non-définis (NULL) 1270
tExternalSortRow 1274
Propriétés du tExternalSortRow 1274
Scénario associé 1275
tExtractDelimitedFields 1276
Propriétés du tExtractDelimitedFields 1276
Scénario : Extraire des champs à partir d'un fichier délimité par des virgules 1277
tExtractPositionalFields 1281
Propriétés du tExtractPositionalFields 1281
Scénario associé 1282
tExtractRegexFields 1283
Propriétés du tExtractRegexFields 1283
Scénario : Extraire des noms, des domaines et domaines de premier niveau à partir d'adresses e-mail 1284
tExtractXMLField 1287
tFilterColumns 1288
Propriétés du tFilterColumns 1288
Scénario associé 1288
tFilterRow 1289
Propriétés du tFilterRow 1289
Scénario : Filtrer une liste de noms et faire une recherche dans la liste 1290
tMap 1294
Propriétés du tMap 1294
Scénario 1 : Mapping simple avec filtre et jointure explicite 1295
Scénario 2 : Mapping avec rejet Inner join 1299
Scénario 3 : Mapping de jointure en cascade 1305
Scénario 4 : Mapping avancé avec filtres, jointures explicites et rejet Inner join 1305
Scénario 5 : Mapping avancé avec filtres et vérification des lignes 1312
Scénario 6 : Mapping avancé avec rechargement de la base de référence à chaque ligne 1316
1323
Scénario 7 : Mapping avec des tables de sortie .1323
tNormalize 1328
Propriétés du tNormalize 1328
Scénario : Normaliser des données 1328
tPerl 1331
Propriétés du tPerl 1331
Scénario : Afficher le nombre de lignes traitées 1331
tPivotToRows 1334
Propriétés du tPivotToRows 1334
Scénario : Concaténer une liste de colonnes à partir d'une table en utilisant les autres colonnes comme pivot 1335
tReplace 1338
Propriétés du tReplace 1338
Scénario : Remplacement mutliple et filtrage de colonnes 1339
tSampleRow 1343
Propriétés du tSampleRow 1343
Scénario : Filtrer des lignes et des groupes de lignes 1343
tSortRow 1346
Propriétés du tSortRow 1346
Scénario : Trier des données 1347

Composants System 1351

tRunJob 1352
Propriétés du tRunJob 1352
Scénario : Exécuter un Job fils 1354
tSetEnv 1359
Propriétés du tSetEnv 1359
Scénario : Modifier la variable *Date* lors de l'exécution d'un Job 1359
tSSH 1363
Propriétés du tSSH 1363
Scénario : Accéder aux informations du système dis-

tant via SSH 1365
tSystem 1367
Propriétés du tSystem 1367
Scénario : Echo 'Hello World!' 1369

Composants Talend MDM 1371

tMDMBulkLoad 1372
Propriétés du tMDMBulkLoad 1372
Scénario : Charger des enregistrements dans l'entité métier ProductFamily 1373
tMDMDelete 1379
Propriétés du tMDMDelete 1379
Scénario : Supprimer des données maître d'un hub MDM 1380
tMDMInput 1384
Propriétés du tMDMInput 1384
Scénario : Lire les données maître d'un hub MDM .. 1385
tMDMOutput 1389
Propriétés du tMDMOutput 1389
Scénario : Ecrire des données maître dans un hub MDM 1391
tMDMReceive 1397
Propriétés du tMDMReceive 1397
Scénario associé 1398
tMDMRouteRecord 1399
Propriétés du tMDMRouteRecord 1399
Scénario : Router un enregistrement vers l'Event Manager 1400
Prérequis du scénario 1400
Router un enregistrement afin de déclencher le processus correspondant 1401
tMDMSP 1410
Propriétés du tMDMSP 1410
Scénario : Exécuter une procédure stockée dans le hub MDM 1411
tMDMViewSearch 1417
Propriétés du tMDMViewSearch 1417
Scénario : Récupérer des enregistrements d'un hub MDM hub via une vue existante 1419
tStewardshipTaskInput 1423
Propriétés du tStewardshipTaskInput 1423
Scénario : Lire des données dans la base de données de la console de stewardship 1424
tStewardshipTaskOutput 1428
Propriétés du tStewardshipTaskOutput 1428
Scénario : Ecrire des enregistrements de données dans la base de données de la console de stewardship 1430

Composants XML 1439

tAdvancedFileOutputXML 1440
Propriétés du tAdvancedFileOutputXML 1440
Définir un arbre XML 1442
Mapping de données XML 1443
Définir le statut du noeud 1444
Scénario : Créer un fichier XML à l'aide d'une boucle 1446
tDTDValidator 1451
Propriétés du tDTDValidator 1451
Scénario : Valider un fichier XML 1451
tExtractXMLField 1454
Propriétés du tExtractXMLField 1454
Scénario 1 : Extraire les données XML d'un champ d'une table de base de données 1455
Scénario 2 : Extraire les données valides et les données erronées à partir d'un champ XML dans fichier délimité 1458
tFileInputXML 1462
Propriétés du tFileInputXML 1462
Scénario 1 : Extraire des adresses XML 1464
Scénario 2 : Extraire les données XML erronées dans un flux de rejet 1466
tFileOutputXML 1470
Propriétés de tFileOutputXML 1470
Scénario : Utiliser un format de sortie XML 1471
tWriteXMLField 1472
Propriétés du tWriteXMLField 1472
Scénario : Extraire la structure d'un fichier XML et l'insérer dans les champs d'une base de données 1473
tXSDValidator 1478
Propriétés du tXSDValidator 1478
Scénario : Valider des flux de données par rapport à un fichier XSD 1479
tXSLT 1482
Propriétés du tXSLT 1482
Scénario : Transformer un fichier XML en HTML à l'aide d'une feuille de style XSL 1482

Préface

Objectif

Ce Guide de référence détaille les principaux composants accessibles à partir de chacun des différents groupes de la **Palette** de **Talend Open Studio**.

Le Guide de référence des composants de **Talend Open Studio** est la version française non exhaustive du **Components Reference Guide** anglais. Pour les composants non documentés, référez-vous à ce dernier. Veillez à consulter régulièrement le site Internet de **Talend** pour télécharger la dernière version du **Guide de référence** français.

Les informations contenues dans ce document s'appliquent aux versions **4.1.x** de **Talend Open Studio**.

Public visé

Ce guide s'adresse aux utilisateurs et aux administrateurs de **Talend Open Studio**.

Dans ce document, l'interface présentée par le biais des captures d'écran peut être légèrement différente de la vôtre.

Conventions typographiques

Ce guide utilise les conventions typographiques suivantes :

- texte en **gras :** boutons et champs d'une fenêtre ou d'une boîte de dialogue, touches du clavier, menus et options des menus ;
- texte en **[gras]:** titre des fenêtres, assistants et boîtes de dialogue ;
- texte en `courier` : paramètres système renseignés par l'utilisateur ;
- texte en *italique* : nom des fichiers, schémas, colonnes, lignes ou variables dont il est question dans les exemples utilisés, ainsi que le nom des champs des vues Basic settings et Advanced settings dont il est fait mention dans le tableau des propriétés de chaque composant.
- Dans la section de propriétés des composants, une icône ou indique si le composant est disponible en Java et/ou en Perl.
- L'icône introduit un paragraphe apportant une information supplémentaire sur un point important. Elle introduit également les commentaires relatifs à un tableau ou une image.
- L'icône introduit un message apportant une information relative aux modalités d'exécution, ou des recommandations particulières. Elle est également utilisée pour attirer l'attention de l'utilisateur sur certaines informations ou situations particulièrement importantes.

Historique des versions

Le tableau ci-dessous liste les modifications apportées au Guide de Référence de **Talend Open Studio**.

Version	Date	Historique
v4.0_a	23/04/2010	Les mises à jour dans le Guide de référence **Talend Open Studio** comprennent : -Nouveaux composants dans les familles File, Databases, Business et Data quality. -Modifications dans les paramètres et scénarios de nombreux composants afin de correspondre aux modifications de l'interface. -Modifications dans le tMap + un nouveau scénario. - Suppression du chapitre Multischema du book et ajout des composants multischema dans le fichier File.
v4.0_b	31/05/2010	Les mises à jour du Guide de Référence **Talend Open Studio** comprennent : -De nouveaux composants dans les chapitres File, Databases, Business, Internet et MDM. -Les composants EXist ont été ajoutés au chapitre Databases. -Des modifications ont été apportées aux propriétés et aux scénarios de nombreux composants, afin de correspondre aux changements de l'interface.
v4.1_a	05/10/2010	Les mises à jour du Guide de Référence **Talend Open Studio** comprennent : -De nouveaux composants dans les chapitres File, Databases, Business, Internet et MDM. -Les variables globales ont été ajoutées au chapitre Orchestration. -Des modifications ont été apportées aux propriétés et aux scénarios de nombreux composants, afin de correspondre aux changements de l'interface.
v4.1_b	13/12/2010	Les mises à jour du Guide de Référence **Talend Open Studio** comprennent : -De nouveaux composants dans les chapitres Business Intelligence, Databases, File, FileScale et Internet. -Les composants Palo ont été ajoutés au chapitre Business Intelligence. -Les composants Microsoft ont été ajoutés au chapitre Internet. -Des modifications ont été apportées aux propriétés et aux scénarios de nombreux composants, afin de correspondre aux changements de l'interface.

Remarques et Support

Votre opinion nous intéresse, n'hésitez pas à nous faire part de vos remarques, suggestions, requêtes concernant la documentation ou le produit, et obtenez le support de notre équipe **Talend**, sur le Forum **Talend** (en anglais) disponible à l'adresse :

http://talendforge.org/forum

Composants Business

Ce composant passe en revue les principaux composants présents dans le groupe **Business** de la **Palette** de **Talend Open Studio**.

La famille **Business** regroupe les connecteurs couvrant des besoins sécifiques, tels que la lecture et l'écriture de base de données de type CRM ou ERP et la lecture et l'écriture de système SAP.

tAlfrescoOutput

Propriétés du tAlfrescoOutput

Famille du composant	Business	
Fonction	Le composant **tAlfrescoOutput** permet de créer des documents dématérialisés sur un serveur Alfresco où ils sont indexés selon des modèles significatifs.	
Objectif	Ce composant permet de créer et de gérer des documents sur un serveur Alfresco.	
Basic settings	*URL*	Saisissez l'URL pour vous connecter à l'application Web Alfresco.
	Login et Password	Entrez les données d'identification de l'utilisateur au serveur Alfresco.
	Base	Saisissez le chemin d'accès de l'emplacement où stocker le document, ou Cochez la case **Map...** puis choisissez, à partir de la liste déroulante **Column**, l'emplacement d'arrivée. Lorsque vous entrez le nom de l'emplacement, assurez-vous de le protéger par des doubles barres obliques inversées (\\).
	Document Mode	Sélectionnez, à partir de la liste déroulante, le mode que vous voulez utiliser pour créer votre document. **Create only** : permet de créer un document s'il n'existe pas préalablement. Notez cependant qu'un message d'erreur apparaîtra si vous essayez de créer un document qui existe déjà. **Create or update** : permet de créer un document qui n'existe pas encore ou de mettre à jour un document existant.
	Container Mode	Sélectionnez, à partir de la liste déroulante, le mode que vous voulez utiliser pour le dossier de destination sur le serveur Alfresco. **Update only** : permet de mettre à jour un dossier de destination déjà existant. Notez cependant qu'un message d'erreur apparaîtra si vous essayez de mettre à jour un document qui n'existe pas. **Create or update** : permet de créer un dossier de destination s'il n'existe pas au préalable, ou de mettre à jour un dossier existant.

	Define Document Type	Cliquez sur les trois points [...] pour afficher l'éditeur **tAlfrescoOutput**. Cet éditeur vous permet : - de sélectionner le fichier dans lequel vous avez défini les métadonnées pour servir de cadre lors de l'enregistrement du document sur Alfresco - de définir le type du document - de sélectionner, à partir de la liste déroulante **available aspects**, l'un des aspects du modèle, puis de cliquer sur le bouton [+] pour ajouter cet aspect à la liste de gauche.
	Property Mapping	Affiche les paramètres renseignés dans l'éditeur **tAlfrescoOutput**, selon lesquels le document sera créé sur le serveur Alfresco. Notez que vous pouvez modifier n'importe lequel des schémas d'entrée de la zone **Property Mapping**.
	Schema et Edit schema	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour effectuer des changements sur le schéma. Notez que le schéma devient automatiquement local (Built-in) lorque vous effectuez des changements.
	Result Log File Name	Permet de parcourir le fichier dans lequel vous voulez enregistrer les logs liés à l'exécution du Job.
Advanced settings	*Configure Target Location Container*	Permet de configurer le type par défaut des conteneurs. Cochez cette case pour afficher de nouveaux champs dans lesquels vous pouvez modifier le type du conteneur et utiliser les types que vous avez créés sur le modèle père/fils.
	Configure Permissions	Lorsque cette case est cochée, l'option permet de configurer manuellement les droits d'accès aux conteneurs et aux documents. Cochez la case **Inherit Permissions** pour synchroniser les droits d'accès entre les conteneurs et les documents. Cliquez sur le bouton [+] pour ajouter de nouvelles lignes à la liste **Permissions** ; vous pourrez alors assigner des rôles à un utilisateur ou groupe d'utilisateurs dans les colonnes User or group et User or group column.
	Encoding	Sélectionnez le type d'encodage à partir de la liste ou choisissez l'option **Custom** pour le définir manuellement. Ce champ doit obligatoirement être renseigné.

	Association Target Mapping	Permet de créer de nouveaux documents sur Alfresco qui contiennent des liens associés à des documents déjà existants sur ce serveur, pour faciliter le processus de navigation par exemple. Pour créer des associations : -Ouvrez l'éditeur **tAlfresco**. -Cliquez sur le bouton **Add** puis sélectionnez un modèle pour lequel vous avez déjà défini des aspects contenant des associations. -Cliquez sur la liste déroulante, en haut de la fenêtre de l'éditeur, et sélectionnez le type de document correpondant. -Cliquez sur **OK** pour fermer l'éditeur et afficher l'association ainsi créée dans la liste **Association Target Mapping**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé en tant que composant de sortie et requiert un composant de début.	
Limitation/Conditions préalables	Pour pouvoir utiliser le composant **tAlfrescoOutput**, il est nécessaire d'installer certaines ressources utiles. Pour plus d'informations, consultez la sous-section *Procédure d'installation*.	

Procédure d'installation

Pour pouvoir utiliser le composant **tAlfrescoOutput** dans Talend Open Studio, vous devez tout d'abord installer le serveur Alfresco ainsi que quelques ressources utiles.

Les sous-sections ci-après indiquent en détail les conditions préalables nécessaires, ainsi que la procédure d'installation.

Conditions préalables

Commencez par exécuter les opérations suivantes :

- Téléchargez le fichier `alfresco-community-tomcat-2.1.0.zip`
- Dézippez-le dans un dossier d'installation, par exemple : `C:\Program Files\Java\jdk1.50_16`
- Installez JDK 1.5.0+
- Effectuez une mise à jour de la variable `JAVA_HOME (JAVA_HOME= C:\alfresco)`
- A partir du dossier d'installation (`C:\alfresco`), lancez le serveur Alfresco en utilisant le script `alf_start.bat`

Assurez-vous que le serveur Alfresco est bien lancé avant d'utiliser le composant **tAlfrescoOutput**.

Installation du module Talend Alfresco

Notez que le fichier `talendalfresco_20081014.zip` est proposé avec le composant **tAlfrescoOutput** dans Talend Open Studio.

Pour installer le module Talend Alfresco :

- A partir du fichier `talendalfresco_20081014.zip`, dans le dossier `talendalfresco_20081014\alfresco`, recherchez les fichiers JAR suivants : `stax-api-1.0.1.jar, wstx-lgpl-3.2.7.jar`, `talendalfresco-client_1.0.jar` et `talendalfresco-alfresco_1.0.jar`, puis déplacez-les vers `C:\alfredsco\tomcat\webapps\alfresco\WEB-INF\lib`
- Ajoutez le filtre d'identification des commandes au fichier `web.xml` situé au chemin d'accès `C:\alfredsco\tomcat\webapps\alfresco\WEB-INF son WEB-INF/`, d'après le modèle de l'exemple situé dans le dossier `talendalfresco_20081014/alfresco` du fichier zippé `talendalfresco_20081014.zip`

Les captures d'écran ci-dessous montrent la partie à ajouter (lignes en bleu) au fichier `web.xml alfresco.`

```
<filter>
   <filter-name>Admin Authen
   <filter-class>org.alfresc
</filter>

<filter-mapping>
   <filter-name>Authenticati
   <url-pattern>/faces/*</ur
</filter-mapping>
```

```
<filter>
   <filter-name>Admin Authentication Filter</filter-name>
   <filter-class>org.alfresco.web.app.servlet.AdminAuthenticationFilter</filter-class>
</filter>

<!-- [talendalfresco] command servlet auth filter def -->
<filter>
   <filter-name>Command Authentication Filter</filter-name>
   <filter-class>fr.openwide.talendalfresco.rest.server.CommandAuthenticationFilter</filter-class>
</filter>

<filter-mapping>
   <filter-name>Authentication Filter</filter-name>
   <url-pattern>/faces/*</url-pattern>
</filter-mapping>
```

```
<filter-mapping>
   <filter-name>Admin Authen
   <url-pattern>/faces/jsp/d
</filter-mapping>

<listener>
   <listener-class>org.apach
</listener>
```

```
<filter-mapping>
   <filter-name>Admin Authentication Filter</filter-name>
   <url-pattern>/faces/jsp/dialog/system-info.jsp</url-pattern>
</filter-mapping>

<!-- [talendalfresco] command servlet auth filter mapping -->
<filter-mapping>
   <filter-name>Command Authentication Filter</filter-name>
   <url-pattern>/command/*</url-pattern>
</filter-mapping>

<listener>
   <listener-class>org.apache.myfaces.webapp.StartupServletContextList
</listener>
```

Informations utiles pour une utilisation avancée du composant

Installer de nouveaux types pour Alfresco :

A partir du fichier `package_jeu_test.zip` , dans le dossier `package_jeu_test/fichiers_conf_alfresco2.1`, recherchez les fichiers suivants : `xml H76ModelCustom.xml` (description du modèle), `web-client-config-custom.xml` (interface Web du modèle) et `custom-model-context.xml` (enregistrement du nouveau modèle), puis collez-les dans le dossier suivant :
`C:/alfredsco/tomcat/shared/classes/alfresco/extension`

Dates :

- Les dates doivent correspondre au type **Talend**, `java.util.Date`
- Les colonnes qui ne comportent ni mapping, ni valeur par défaut (celles du type `Date` par exemple) se voient attribuer une chaîne de caractères vide.
  Solution : effacez toutes les colonnes sans mapping ni valeur par défaut. Notez cependant qu'une modification de type Alfresco les rétablira.

Contenu :

- Ne confondez pas le chemin d'accès au fichier dont vous voulez créer le contenu et celui de son emplacement cible sur Alfresco.
- N'oubliez pas l'URL ! Elle permet de cibler de nombreux protocoles, dont le transfert de fichiers et le HTTP.
- Précédez les URLs renvoyant à des fichiers du système de fichiers en réseau de "file:", si vous utilisez Windows en local, ou de "file://", si vous utilisez soit Windows sur un réseau (qui accepte également la mention "file: \ \"), soit Linux.
- N'utilisez jamais la double barre oblique inversée (\\) dans le chemin d'accès cible (échappement automatique), à moins que vous n'entriez le chemin directement dans les propriétés de base (Basic settings) du composant **tAlfrescoOutput** ou que vous n'effectuiez une concaténation, dans l'éditeur **tMap** par exemple.

Propriétés multiples ou associations :

- Il est possible de n'en créer qu'une seule par document en y mappant une valeur String, ou une ou plusieurs en y mappant une valeur List(objet).
- Il est possible de vider une propriété multiple en lui mappant une liste vide que vous pouvez créer, par exemple, en utilisant la commande `new java.util.ArrayList()` dans le composant **tMap**.
- En revanche, il est impossible de supprimer une association.
- Faire des `List(object)` avec le composant **tAggregate** :
  -définissez dans un fichier une table de relation n-n, contenant par exemple une ligne `name` (pouvant être retrouvée dans les imput rows) et une ligne `category` (pouvant être définie avec son mapping dans un fichier tiers).
  -group by : input name, output name.
  -opération: output `categoryList`, fonction `list(object)`, input `category`. ATTENTION il s'agit d'une list (object) et non d'une simple list !

Références (documents et dossiers) :

- Les références sont créées par mapping d'une ou plusieurs références de noeud existantes (xpath ou namepath), en type `String` ou `List(object).`
- Une erreur dans l'association ou la propriété de type référence n'empêche pas la création du noeud qui la porte.
- Les propriétés de type référence sont créées dans l'onglet des propriétés standards (**Basic Settings**).
- Les associations sont créées dans l'onglet des propriétés avancées (**Advanced Settings**).

Dématérialisation, tAlfrescoOutput et Gestion de Contenu

La dématérialisation est le processus qui sert à convertir les documents concrets au format numérique, permettant ainsi de passer d'une utilisation de documentation physique à des systèmes de Gestion de Contenu électroniques. Un système de Gestion de Contenu permet de gérer un large éventail de documents, allant des plus basiques aux certificats d'actions par exemple.

Les entreprises procèdent à la dématérialisation de leur contenu par traitement des documents, qui peut être réalisé soit à la main, soit de manière automatique grâce à une machine

Compte tenu de la diversité des contenus à dématérialiser, les entreprises sont obligées d'utiliser des technologies diverses. Scanner les documents papier, créer des interfaces pour capter les documents électroniques d'autres applications, convertir les documents images en documents textes lisibles et modifiables par une machine... Ceci n'est qu'un exemple des technologies employées.

De plus, les documents scannés ainsi que les fax numériques ne peuvent être lus par des machines. Différentes technologies de reconnaissance des caractères sont utilisées pour les convertir en données utilisables. La Reconnaissance de Caractères Manuscrits (RCM) et la Reconnaissance Optique de Marques (ROM) en sont deux exemples.

Les métadonnées de support, qui permettent d'identifier le contenu de manière efficace grâce à des requêtes spécifiques, sont au moins aussi importantes que le contenu capté lors de la dématérialisation, en différents formats et à partir de nombreuses sources.

Dès lors, comment agréger et indexer dans un système de Gestion de Contenu le contenu d'un document en même temps que ses métadonnées associées, pour qu'il puisse être extrait et traité de manière significative ? **Talend** vous apporte la réponse par le biais de son composant **tAlfrescoOutput**.

Le composant **tAlfrescoOutput** vous permet de stocker et de gérer vos documents électroniques, ainsi que les métadonnées qui y sont associées, sur le serveur Alfresco, système leader sur le marché de la Gestion de Contenu.

La figure ci-dessous illustre le rôle de **Talend** à la fois dans le processus de dématérialisation et dans le système Gestion de Contenu (Alfresco).

Scénario : Créer des documents sur un serveur Alfresco

Ce scénario Java décrit un Job à deux composants permettant de créer deux fichiers de documents et leurs métadonnées associées sur un serveur Alfresco, le système de Gestion de Contenu en Java.

- A partir de la Palette, cliquez-déposez les composants **tFileInputDelimited** et **tAlfrescoOutput** sur l'éditeur graphique.
- Reliez ces deux composant à l'aide d'un lien **Row Main**.

- Dans l'éditeur graphique, double-cliquez sur le **tFileInputDelimited** pour en afficher les paramètres de base (**Basic settings**).
- Renseignez le chemin d'accès au fichier dans le champ **File Name** ainsi que toutes les autres propriétés. Notez cependant que si vous avez déjà stocké vos schémas d'entrée en local dans le **Repository**, vous pouvez tout simplement récupérer le fichier choisi à partir du noeud **Metadata** et le déposer sur l'éditeur graphique. Les paramètres du fichier délimité s'afficheront alors automatiquement dans les champs concernés de l'onglet **Basic settings** du composant concerné.
  Pour plus d'informations sur les métadonnées, consultez la section *Configurer un schéma File Delimited* du **Guide Utilisateur** de **Talend Open Studio**.

Dans ce scénario, le fichier délimité présente les métadonnées et le chemin d'accès de deux documents que vous allez créer sur le serveur Alfresco. Le schéma d'entrée des documents de décompose en quatre colonnes : `file_name, destination_folder name, source_path,` et `author`.

	A	B	C	D	E	F
1	file_name;destination_folder_name;source_path;author					
2	PO_43278.pdf;FINANCE\PO;file:D:/PO/customer1_2009-03-24.pdf;talend					
3	PO_43279.pdf;FINANCE\PO;file:D:/PO/customer2_2009-03-24.pdf;talend					

Le schéma d'entrée du fichier délimité sera donc comme suit :

- Dans l'éditeur graphique, double-cliquez sur le composant **tAlfrescoOutput** pour en afficher les paramètres de base (Basic settings)**.**

- Dans la partie **Alfresco Server**, saisissez l'URL du serveur Alfresco ainsi que les données d'identification de l'utilisateur dans les champs correspondants.

- Dans la partie **TargetLocation**, vous pouvez soit saisir le nom de l'emplacement du serveur où vous voulez stocker le document, soit cocher la case **Map...** puis choisir l'emplacement cible à partir de la liste déroulante **Column**, soit `destination_folder_name` dans ce scénario.

  Lorsque vous renseignez le nom de l'emplacement, assurez vous de bien utiliser le caractère d'échappement double barre oblique inversée (\\).

- Dans la liste déroulante **Document Mode**, sélectionnez le mode que vous voulez utiliser pour la création de vos documents.
- Dans la liste déroulante **Container Mode**, sélectionnez le mode que vous voulez utiliser pour le dossier de destination sur Alfresco.
- Cliquez sur le bouton [...] de l'option **Define Document Type** pour ouvrir l'éditeur **tAlfrescoOutput**.

- Cliquez sur le bouton **Add** pour parcourir et sélectionner le fichier xml qui comprend les métadonnées selon lesquelles vous voulez sauvegarder les documents sur Alfresco.
  Tous les aspects disponibles du modèle sélectionné s'affichent dans la liste **Available Aspects**.

  Vous pouvez parcourir le dossier modèle en local ou en réseau. Une fois définis les aspects à appliquer au document que vous allez créer sur Alfresco, vous n'aurez plus besoin de ce dossier.

- Si nécessaire, sélectionnez le ou les aspect(s) à ajouter aux métadonnées à écrire sur le serveur Alfresco, à partir de la liste déroulante **Available Aspects**. Dans ce scénario, l'objectif est de faire en sorte que le nom de l'auteur fasse partie des métadonnées enregistrées sur Alfresco.
- Cliquez sur la liste déroulante, située en haut de la fenêtre de l'éditeur, pour sélectionner le type à donner aux documents créés sur Alfresco. Dans cet exemple, il s'agit de `Content`. Tous les aspects définis, utilisés pour sélectionner les métadonnées à écrire sur Alfresco, s'affichent dans la liste **Property Mapping** de l'onglet **Basic Settings** du composant **tAlfrescoOutput**. Dans ce scénario, il s'agit de trois aspects, deux aspects de base pour le type `Content` (`content` et `name`) ainsi qu'un aspect aditionnel (`author`).
- Cliquez sur **Sync columns** pour propager automatiquement toutes les colonnes du fichier délimité.
- Si nécessaire, cliquez sur **Edit schema** pour avoir un aperçu de la structure des données du **tAlfrescoOutput** en sortie.

- Cliquez sur le bouton [...] situé à côté du champ **Result Log File Name** puis parcourez le fichier où vous voulez sauvegarder les logs après exécution du Job.
- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

Les deux documents sont créés sur Alfresco grâce aux métadonnées des schémas d'entrée.

tBonitaDeploy

Propriétés du tBonitaDeploy

Famille de composant	Business/Bonita	
Fonction	Le **tBonitaDeploy** configure un moteur Bonita Runtime et déploie un processus spécifique Bonita (un fichier .bar exporté de la solution Bonita solution) vers ce moteur.	
Objectif	Ce composant déploie un processus spécifique Bonita vers Bonita Runtime.	
Basic settings	*Bonita Runtime Environment File*	Parcourez votre répertoire, ou saisissez le chemin d'accès au fichier d'environnement Bonita Runtime.
	Bonita Runtime Jass File	Parcourez votre répertoire, ou saisissez le chemin d'accès au fichier jass Bonita Runtime.
	Bonita Runtime logging file	Parcourez votre répertoire, ou saisissez le chemin d'accès au fichier de log Bonita Runtime.
	Business Archive	Parcourez votre répertoire, ou saisissez le chemin d'accès au fichier .bar Bonita de traitement que vous souhaitez utiliser.
	User name	Saisissez votre identifiant de connexion à Bonita.
	Password	Saisissez votre mot de passe de connexion à Bonita.
	Die on error	Cette case n'est pas cochée par défaut, pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les métadonnées de traitement du Job au niveau du Job ainsi qu'au niveau de chaque composant.
Utilisation	Ce composant est généralement utilisé en standalone.	
Connections		Liens de sortie (d'un composant à un autre) : **Trigger** : Run if, OnComponentOk, OnComponentError, OnSubjobOk, OnSubjobError. Liens d'entrée (d'un composant à un autre) : **Trigger** : Run if, OnComponentOk, OnComponentError, OnSubjobOk, OnSubjobError Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Global Variables		**Process Definition UUID** : Indique le numéro d'ID du processus déployé. Disponible en tant que variable **Flow**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur** **Talend Open Studio**.

Limitation	Le fichier d'environnement Bonita Runtime, le fichier jaas Bonita Runtime et le fichier de log Bonita Runtime doivent être stockés sur le serveur d'exécution du Job, à l'aide de ce composant.

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tBonitaInstantiateProcess

tBonitaInstantiateProcess Properties

Famille de composant	Business/Bonita	
Fonction	Le composant **tBonitaInstantiateProcess** instancie un processus existant dans un moteur Bonita Runtime puis lance l'exécution.	
Objectif	Ce composant commence l'instance d'un processus spécifique déployé dans le moteur Bonita Runtime.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champ qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.. Pour ce composant, le schéma dépend du **Module** sélectionné.
	Bonita Runtime Environment File	Parcourez votre répertoire, ou saisissez le chemin d'accès au fichier d'environnement Bonita Runtime.
	Bonita Runtime Jass File	Parcourez votre répertoire, ou saisissez le chemin d'accès au fichier jass Bonita Runtime.
	Bonita Runtime logging file	Parcourez votre répertoire, ou saisissez le chemin d'accès au fichier de log Bonita Runtime.
	Use Process ID	Cette case n'est pas cochée par défaut, pour activer les champs **process name** (nom du processsus)et **process version** (version du processus), afin que vous puissiez saisir les informations correspondantes du processus spécifique que vous souhaitez instancier. Ces informations seront utimlisées pour générer automatiquement l'ID du processus. Une fois la case cochée, le champ **Process definition ID** est activé, et vous pouvez saisir l'ID requis du processus. L'ID du processus est créé lorsque ce processus est déployé dans le moteur Bonita Runtime.
	User name	Saisissez l'identifiant utilisé pour instancier ce processus..
	Password	Saisissez le mot de passe utilisé pour instancier ce processus.
	Die on error	Cette case n'est pas cochée par défaut, pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les métadonnées de traitement du Job au niveau du Job ainsi qu'au niveau de chaque composant.

Utilisation	Ce composant est généralement utilisé en standalone ou en tant que composant de sortie.	
Connections		Liens de sortie (d'un composant à un autre) : **Trigger** : Run if, OnComponentOk, OnComponentError, OnSubjobOk, OnSubjobError. Liens d'entrée (d'un composant à un autre) : **Row** : Main. **Trigger** : Run if, OnComponentOk, OnComponentError, OnSubjobOk, OnSubjobError Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Global Variables		**Process Definition UUID** : Indique le numéro d'ID du processus déployé. Disponible en tant que variable **Flow**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	n/a	

Scénario : Exécuter un processus Bonita via un Job Talend

Ce scénario décrit un Job déployant un processus Bonita dans le moteur Bonita Runtime et exécutant ce processus, dans lequel une requête sur le personnel est traitée.

Ce Job utilise trois composants.

- Un **tBonitaDeploy** : ce composant déploie un processus Bonita dans le moteur Bonita Runtime.
- Un **tFixedFlowInput** : ce composant génère le schéma utilisé comme paramètre d'exécution du processus déployé.

- Un **tBonitaInstantiateProcess** : ce composant exécute le processus déployé.

> Lorsque vous générez un schéma à l'aide d'un **tFixedFlowInput**, les noms de colonnes du schéma doivent être identiques à ceux des paramètres Bonita utilisés pour exécuter le même processus dans Bonita.

Avant de commencer à reproduire ce schéma, préparez votre fichier Bonita .bar, c'est-à-dire le processus exporté du système Bonita et qui sera déployé dans le moteur Bonita Runtime. Dans ce scénario, ce fichier est *TEST--4.0.bar*. Ce processus peut être vérifié via l'interface Bonita.

Pour reproduire ce scénario, procédez comme suit :

- Déposez les composants **tBonitaDeploy**, **tFixedFlowInput** et **tBonitaInstantiateProcess** de la **Palette** dans l'espace de modélisation graphique.
- Cliquez-droit sur le composant **tBonitaDeploy** pour ouvrir son menu contextuel.
- Sélectionnez **Trigger** > **On Subjob Ok** pour relier le **tBonitaDeploy** au **tFixedFlowInput**.
- Cliquez-droit sur le **tFixedFlowInput** pour ouvrir son menu contextuel et sélectionnez **Row** > **Main** pour relier ce composant au **tBonitaInstantiateProcess**, à l'aide d'un lien **Main**.
- Double-cliquez sur le **tBonitaDeploy** afin d'ouvrir sa vue **Basic settings**.

- Dans la zone **Bonita Runtime Configuration**, parcourez votre répertoire de fichiers de variables Bonita Runtime.
  Dans le champ **Bonita Runtime Environment file**, parcourez votre répertoire jusqu'au fichier *bonita-environnement.xml*.
  Dans le champ **Bonita Runtime Jaas File**, parcourez votre répertoire jusqu'au fichier *jaas-standard.cfg*.
  Dans le champ **Bonita Runtime Logging File**, parcourez votre répertoire jusqu'au fichier *logging.properties*.
- Dans le champ **Business Archive**, parcourez votre répertoire jusqu'au fichier Bonita .bar, qui est le processus exporté de votre système Bonita et sera déployé dans le moteur Bonita Runtime.
- Dans les champs **Username** et **Password**, saisissez vos informations de connexion à Bonita.
- Double-cliquez sur le composant **tFixedFlowInput** afin d'ouvrir sa vue **Basic settings**.

- Cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour ouvrir l'éditeur du schéma.

- Dans l'éditeur du schéma, cliquez sur le bouton **[+]** pour ajouter une ligne.

- Cliquez sur la nouvelle ligne et saisissez son nom : *name*.
- Cliquez sur **OK**.
- Dans la zone **Mode** de la vue **Basic settings**, sélectionnez l'option **Use inline table**.
- Sous le tableau, cliquez sur le bouton **[+]** afin d'ajouter une ligne.
- Dans le tableau, cliquez sur la ligne ajoutée et saisissez entre guillemets le nom de la personne de votre personnel : *ychen*, dont la requête sera traitée par le processus déployé.
- Double-cliquez sur le **tBonitaInstantiateProcess** afin d'ouvrir sa vue **Basic settings**.

- Dans la vue **Basic settings**, cliquez sur le bouton **Sync columns** afin de récupérer le schéma du composant précédent.
- Dans la zone **Bonita Runtime Configuration**, parcourez votre répertoire de fichiers de variables Bonita Runtime.
  Dans le champ **Bonita Runtime Environment file**, parcourez votre répertoire jusqu'au fichier *bonita-environnement.xml*.
  Dans le champ **Bonita Runtime Jaas File**, parcourez votre répertoire jusqu'au fichier *jaas-standard.cfg*.
  Dans le champ **Bonita Runtime Logging File**, parcourez votre répertoire jusqu'au fichier *logging.properties*.
- Cochez la case **Use Process ID** pour activer le champ **Process Definition Id**.
- Dans le champ **Process Definition Id**, cliquez entre les guillemets et appuyez sur **Ctrl+Espace** afin d'ouvrir la liste d'autocomplétion contenant les variables globales disponibles pour ce Job.
- Double-cliquez sur la variable que vous souhaitez utiliser pour l'ajouter entre les guillemets. Dans ce scénario, double-cliquez sur *tBonitaDeploy_1_ProcessDefinitionUUID*, qui récupère l'ID de définition du processus déployé par le composant **tBonitaDeploy**.

Si le processus a été déployé et que le **tBonitaDeploy** n'est pas utilisé, décochez la case **Use Process ID** afin d'activer les champs **Process name** et **Process version** et remplir les informations correspondantes. Le **tBonitaInstantiateProcess** concatène le nom et la version du processus que vous avez saisis lors de la construction de l'ID du processus.

- Dans les champs **Username** et **Password**, saisissez respectivement votre identifiant et votre mot de passe de connexion à Bonita.
- Appuyez sur **F6** pour exécuter le Job.

Le processus est déployé dans le moteur Bonita Runtime et une instance est créée pour les requêtes concernant le personnel.

tCentricCRMInput

Propriétés du tCentricCRMInput

Famille du composant	Business/CentricCRM	
Fonction	Le composant **tCentricCRMInput** se connecte à un module de la base de données CentricCRM via le service Web adéquat.	
Objectif	Le **tCentricCRMInput** permet d'extraire les données d'une base de données Centric CRM à l'aide d'une requête.	
Basic settings	*CentricCRM URL*	Saisissez l'URL du service Web permettant de se connecter à la base de données CentricCRM.
	Module	Sélectionnez le module adéquat dans la liste.
	Server	Saisissez l'adresse IP du serveur de base de données.
	UserID et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champ qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.. Pour ce composant, le schéma dépend du **Module** sélectionné.
	Query condition	Saisissez la requête permettant de sélectionner les données à extraire.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les métadonnées de traitement du Job au niveau du Job ainsi qu'au niveau de chaque composant.
Utilisation	Ce composant est généralement utilisé en tant que composant de début. Un composant de sortie est nécessaire.	
Limitation	n/a	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tCentricCRMOutput

Propriétés du tCentricCRMOutput

Famille du composant	Business/CentricCRM	
Fonction	Le composant **tCentricCRMOutput** écrit dans un module de la base de données CentricCRM via le service Web adéquat.	
Objectif	Le **tCentricCRMOutput** permet d'écrire des données dans une base de données CentricCRM.	
Basic settings	*CentricCRM URL*	Saisissez l'URL du service Web permettant de se connecter à la base de données CentricCRM.
	Module	Sélectionnez le module adéquat dans la liste.
	Server	Saisissez l'adresse IP du serveur de base de données.
	UserID et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Action	Vous pouvez effectuer l'une des opérations suivantes sur les données du module CentricCRM : **Insert** : insérer les données. **Update** : mettre les données à jour. **Delete** : supprimer les données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champ qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.. Pour ce composant, le schéma dépend du **Module** sélectionné.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les métadonnées de traitement du Job au niveau du Job ainsi qu'au niveau de chaque composant.
Utilisation	Ce composant est utilisé comme composant de sortie. Il nécessite un composant d'entrée.	
Limitation	n/a	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tHL7Input

Propriétés du tHL7Input

Famille de composant	Business	
Fonction	Lit un fichier de structure HL7 et extrait des données ligne par ligne.	
Objectif	Ce composant lit un fichier HL7 structuré ou un flux de données ligne par ligne pour le scinder en champs et envoie les champs tels que définis dans le schéma au composant suivant du Job via une connexion **Row**.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier où sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant connexion de base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Multi Schemas Editor	L'éditeur **[Multi Schema Editor]** vous permet de construire et de configurer le flux de données dans un fichier délimité multi-structure, pour associer un schéma par sortie.
	Segment Lists	**Connection** : Les colonnes sont automatiquement récupérées à partir du fichier d'entrée. Le nom de la colonne est le nom du segment. **Column Mapping** :. Dans ce tableau, le mapping est récupéré de celui que vous avez défini dans l'éditeur.
	Not Validate HL7 Message	Cochez cette case si vous ne souhaitez pas valider les messgaes HL7.
Advanced settings	*Advanced separator (for numbers)*	Cochez cette option pour modifier les séparateurs utilisés pour les nombres : **Thousands separator** : définissez le séparateur utilisé pour les milliers. **Decimal separator** : définissez le séparateur utilisé pour les décimaux.
	Encoding	Sélectionnez le type d'encodage à partir de la liste ou choisissez l'option **Custom** pour le définir manuellement. Ce champ doit obligatoirement être renseigné.

	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Généralement utilisé en tant que composant d'entrée. Un composant de sortie est requis.	
Limitation	n/a	

Scénario : Retrouver des informations concernant les patients et les événements à partir d'un fichier HL7

Ce scénario décrit un Job de quatre composants, dont le but est de retrouver des informations concernant des patients et des événements à partir d'un fichier HL7.

- A partir de la **Palette**, glissez-déposez un composant **tHL7Input**, et trois **tLogRow** dans l'espace de modélisation graphique.
- Double-cliquez sur le composant **tHL7Input** afin d'ouvrir son éditeur.

- Dans le champ **File path**, cliquez sur **[Browse...]**, parcourez votre répertoire, et sélectionnez votre fichier HL7.
- Dans la zone **File Setting**, saisissez le caractère de début de votre segment (**Start character**), et le caractère de fin de votre segment (**End character**).
- Dans la zone **Schema View**, dans la liste **Segment(As Schema)**, sélectionnez **MSH**.
- Glissez-déposez les segments *MSH-3(1)[HD]* et *MSH-7(1)[TS]* de la zone **Message View** dans la zone **Schema View**.

- Dans la zone **Schema View**, dans la liste **Segment(As Schema)**, sélectionnez **EVN**.
- Glissez déposez les segments *EVN-1(1)-1-1[ID]* et *EVN-2(1)-1-1ST* de la zone **Message View** dans la zone **Schema View**.

- Dans la zone **Schema View**, dans la liste **Segment(As Schema)**, sélectionnez **PID**.
- Glissez-déposez les segments suivants de la zone **Message View** dans la zone **Schema View** : *PID-1(1)-1-1[SI], PID-5(1)-1-1ST, PID-5(1)-2-1ST, PID-5(1)-3-1ST, PID-5(1)-4-1ST, PID-5(1)-5-1ST, PID-5(1)-7-1[ID]*.

  S'il est disponible, utilisez le bouton **Auto map!** situé en bas à gauche de la fenêtre, il permet d'effectuer l'opération de mapping automatiquement.

- Cliquez sur **Ok** afin de fermer l'éditeur.
- Reliez le **tHL7Input** aux trois **tLogRow**, en utilisant respectivement les liens **MSH**, **EVN** et **PID**.
- Sauvegardez votre Job, et appuyez sur **F6** pour l'exécuter.

```
Starting job HL7 at 10:15 02/04/2010.

.---------+------------.
|      tLogRow_1        |
|=--------+-----------=|
|MSH_LABEL|MSH_DATE    |
|=--------+-----------=|
|MedSeries|200903230934|
'---------+------------'

.------+---------+---------+------------+-----------+---------+--------+----------.
|                                   tLogRow_2                                      |
|=-----+---------+---------+------------+-----------+---------+--------+---------=|
|PID_ID|PID_FNAME|PID_LNAME|PID_STNUMBER|PID_STREET |PID_CITY |PID_BDAY|PID_GENDER|
|=-----+---------+---------+------------+-----------+---------+--------+---------=|
|1     |Marie    |Charlotte|456         |main_street|Richemont|19770202|F         |
'------+---------+---------+------------+-----------+---------+--------+----------'
.------+-----------.
|     tLogRow_3     |
|=-----+-----------=|
|EVN_ID|EVN_DATE   |
|=-----+-----------=|
|A31   |200903230934|
'------+-----------'
Job HL7 ended at 10:15 02/04/2010. [exit code=0]
```

La console affiche les trois tables des **tLogRow**, qui retournent différents types d'informations. La première montre l'en-tête du message et sa date. La seconde retourne les informations concernant le patient. La troisième affiche l'ID de l'événement ainsi que sa date.

tHL7Output

Propriétés du tHL7Output

Famille de composant	Business	
Fonction	Ecrit un fichier de structure HL7 et insère des données ligne par ligne.	
Objectif	Ce composant écrit un fichier HL7 structuré selon la norme HL7.	
Basic settings	*Property type*	Peut être Built-In ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier où sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Schema(s)	**Schema** : Définissez le noeud où stocker les données de la ligne parente. **Parent row** : Flux de données source.
	File Name	Parcourez le répertoire où stocker les fichiers générés.
	Configure HL7 Tree	Ouvre l'interface dédiée pour vous aider à mettre en place le mapping HL7.
	HL7 version	Sélectionnez dans la liste votre version de HL7.
Advanced settings	*Create directory only if not exists*	Cette case est cochée par défaut. Cette option permet de créer le dossier contenant le fichier de sortie s'il n'existe pas déjà.
	Encoding	Sélectionnez le type d'encodage à partir de la liste ou choisissez l'option **Custom** pour le définir manuellement. Ce champ doit obligatoirement être renseigné.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Généralement utilisé en tant que composant de sortie. Un composant d'entrée est requis.	
Limitation	n/a	

Scénario associé

Pour un scénario utilisant un autre composant HL7, consultez *Scénario : Retrouver des informations concernant les patients et les événements à partir d'un fichier HL7, page xlix*.

tMicrosoftCRMInput

Propriétés du tMicrosoftCRMInput

Famille de composant	Business	
Fonction	Le composant **tMicrosoftCRMInput** se connecte à une entité de la base de données MicrosoftCRM via le service Web adéquat.	
Objectif	Ce composant permet d'extraire les données d'une base MicrosoftCRM selon certaines conditions définies dans des colonnes spécifiques.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier où sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Microsoft Webservice URL	Saisissez l'URL du service Web permettant de se connecter à la base de données Microsoft CRM.
	Organizename	Renseignez le nom de l'utilisateur ou de l'organisme, défini par un administrateur, qui veut accéder à la base de données MicrosoftCRM.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Domain	Saisissez le nom de domaine du serveur sur lequel est hébergée la base MicrosoftCRM.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.
	Time out (seconds)	Nombre de secondes d'écoute du port avant sa fermeture.
	Entity	Sélectionnez l'entité adéquate à partir de la liste déroulante.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Si vous effectuez des modifications, le schéma passera automatiquement en built-in. Pour ce composant, le schéma correspond à l'entité sélectionnée.
	Logical operators used to combine conditions	Si vous souhaitez combiner le filtrage simple et le mode avancé, sélectionnez l'opérateur permettant de combiner les deux modes.

	Conditions	Les conditions sont exécutées les unes après les autres pour chaque ligne. **Input column** : Cliquez dans la cellule et sélectionnez la colonne du schéma d'entrée à laquelle attribuer la condition. **Operator** : Cliquez dans la cellule et sélectionnez l'opérateur pour relier la valeur à la colonne du schéma. **Value** : Saisissez la valeur filtrée entre guillemets, si nécessaire. Cliquez sur le bouton **[+]** pour ajouter autant de conditions que vous le souhaitez.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé en tant que composant de début. Un composant de sortie est nécessaire.	
Limitation	n/a	

Scénario : Ecrire des données dans une base de données Microsoft CRM et attribuer des conditions aux colonnes pour extraire des lignes spécifiques

Le scénario suivant décrit un Job à quatre composants qui a pour but l'écriture des données d'un fichier délimité en entrée dans une entité personnalisée d'une base de données MicrosoftCRM . Le Job permet ensuite d'extraire des lignes spécifiques jusque dans un fichier de sortie, en utilisant les conditions définies sur certaines colonnes d'entrée.

Si vous voulez écrire dans une Entité Personnalisée d'une base de données MicrosoftCRM, assurez-vous de nommer les colonnes conformément aux règles de syntaxe Microsoft, à savoir "name_columnname" en minuscules.

- A partir de la **Palette**, déposez les composants suivants dans l'éditeur graphique : **tFileInputdelimited**, **tFileOutputDelimited**, **tMicrosoftCRMInput**, et **tMicrosoftCRMOutput**.

- Connectez le composant **tFileInputDelimited** au **tMicrosoftCRMOutput** à l'aide d'un lien de type **Row Main**.
- Connectez le composant **tMicrosoftCRMIntput** au **tFileOutputDelimited** à l'aide d'un lien de type **Row Main**.
- Connectez le composant **tFileInputDelimited** au **tMicrosoftCRMInput** à l'aide d'un lien de type **OnSubjobOk**.
- Double-cliquez sur le composant **tFileInputDelimited** pour afficher la vue **Basic settings** et configurer ses paramètres de base.

- Sélectionnez le mode **Repository** à partir de la liste déroulante **Property Type** si vous avez stocké les paramètres du fichier d'entrée sous le noeud **Metadata** de l'arborescence **Repository**. Dans le cas contraire, sélectionnez le mode **Built-In** et remplissez manuellement les champs correspondants. Dans cet exemple, les propriétés sont définies en mode **Built-In**.
- Cliquez sur le bouton [...] à côté du champ **File Name/Input Stream** et parcourez vos dossiers jusqu'au fichier délimité contenant les données d'entrée. Dans cet exemple, le fichier d'entrée comprend les colonnes suivantes : *new_id*, *new_status*, *new_firstname*, *new_email*, *new_city*, *new_initial* et *new_zipcode*.

```
new_id;new_status;new_firstname;new_email;new_city;new_initial;new_zipcode
1;married;Paul;pnewman@comp.com;New York;P.M;55677
2;single;Raul;pnewman@comp.com;New York;R.L;55677
3;single;Mary;mnewman@comp.com;Chicago;M.B;66898
4;married;John;Jnewman@comp.com;Chicago,J.M;66898
5;single;Martin;mnewman@comp.com;Sunnyvale,M.P;33662
6;married;Janet;jnewman@comp.com;Sunnyvale,J.P;33662
7;married;Harry;hnewman@comp.com;New York;H.M;55677
8;married;Jerry;jnewman@comp.com;New York;J.M;55677
9;married;Alice;anewman@comp.com;New York;A.M;55677
10;single;Jack;jnewman@comp.com;New York;J.M;55677
```

- Dans le champ **Row separator**, définissez le séparateur de lignes permettant d'identifier les fins de ligne. De même, pour **Field separator**, définissez le séparateur de champs permettant de délimiter les différents champs pour chaque ligne.

- Définissez, si nécessaire, le nombre de lignes d'en-tête (**Header**) et de pied-de-page (**Footer**) à ignorer, ainsi que le nombre maximum de lignes (**Limit**) que vous souhaitez traiter. Dans cet exemple, aucun de ces trois champs ne sont définis.
- Cliquez sur **Edit schema** pour ouvrir une boîte de dialogue dans laquelle vous pouvez définir le schéma d'entrée que vous voulez écrire dans la base de données MicrosoftCRM.

- Cliquez sur **OK** pour fermer la boîte de dialogue.
- Double-cliquez sur le composant **tMicrosoftCRMOutput** pour afficher la vue **Basic settings** et configurer ses paramètres de base.

- Dans le champ **Microsoft Web Service URL**, saisissez l'URL du serveur Web Microsoft. Renseignez ensuite le nom (**Username**) ainsi que le mot de passe (**Password**) correspondants.

- Dans le champ **OrganizeName**, saisissez l'identifiant qui porte les droits d'accès à la base de données MicrosoftCRM.
- Dans le champ **Domain**, renseignez le nom de domaine du serveur sur lequel la base de données MicrosoftCRM est hébergée. Saisissez ensuite l'adresse IP (**Host**) et le numéro de port d'écoute (**Port**) du serveur.
- A partir de la liste déroulante **Action**, sélectionnez l'opération que vous voulez effectuer. Dans cet exemple, le but est d'insérer (**Insert**) des données dans une entité personnalisée sur la base MicrosoftCRM.
- Dans le champ **Time out**, définissez la limite de temps (en secondes) après laquelle le Job s'arrête automatiquement.
- Dans la liste **Entity**, sélectionnez l'une des options proposées. Dans cet exemple, c'est l'option *CustomEntity* qui est sélectionnée.

  Lorsque l'option *CustomEntity* est sélectionnée, le champ **Custom Entity Name** apparaît, dans lequel vous devez saisir le nom de l'entité personnalisée.

- Le **Schema** est alors défini de manière automatique, en fonction de l'entité sélectionnée. Cliquez si nécessaire sur le bouton **Edit schema** pour afficher une boîte de dialogue à partir de laquelle vous pouvez modifier ce schéma et retirer les colonnes dont vous n'avez pas besoin en sortie.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent.

- Double-cliquez sur le composant **tMicrosoftCRMInput** pour afficher la vue **Basic settings** et configurer ses paramètres de base.

- Sélectionnez le mode **Repository** à partir de la liste déroulante **Property Type** si vous avez stocké les paramètres du fichier d'entrée sous le noeud **Metadata** de l'arborescence **Repository**. Dans le cas contraire, sélectionnez le mode **Built-In** et remplissez manuellement les champs correspondants. Dans cet exemple, les propriétés sont définies en mode **Built-In**.
- Dans le champ **Microsoft Web Service URL**, saisissez l'URL du serveur Web Microsoft. Renseignez ensuite le nom (**Username**) ainsi que le mot de passe (**Password**) correspondants.
- Dans le champ **OrganizeName**, saisissez l'identifiant qui porte les droits d'accès à la base de données MicrosoftCRM.
- Dans le champ **Domain**, renseignez le nom de domaine du serveur sur lequel la base de données MicrosoftCRM est hébergée. Saisissez ensuite l'adresse IP (**Host**) et le numéro de port d'écoute (**Port**) du serveur.
- Dans le champ **Time out**, définissez la limite de temps (en secondes) après laquelle le Job s'arrête automatiquement.
- Dans la liste **Entity**, sélectionnez l'une des options proposées. Dans cet exemple, c'est l'option *CustomEntity* qui est sélectionnée.

- Le **Schema** est alors défini de manière automatique, en fonction de l'entité sélectionnée, mais vous pouvez le modifier si nécessaire. Dans cet exemple, pour accéder à l'entité personnalisée, il faut définir le schéma manuellement. Copiez le schéma à sept colonnes à partir du composant **tMicrosoftCRMOutput** et collez-le dans la boîte de dialogue du **tMicrosoftCRMInput**.

- Cliquez sur le bouton **OK** pour fermer la boîte de dialogue. Un message pop-up vous proposera de propager les modifications : cliquez sur le bouton **Yes**.
- Dans la vue **Basic settings**, sélectionnez l'opérateur logique **And** ou **Or** à utiliser pour combiner les conditions définies dans les colonnes d'entrée. Dans cet exemple, pour définir deux conditions sur deux colonnes d'entrée différentes, il faut utiliser l'opérateur logique **And**.
- Dans la zone **Condition**, cliquez sur le bouton **[...]** pour ajouter autant de lignes que nécessaire, puis cliquez dans chaque ligne et sélectionnez, à partir de la liste déroulante **Input column**, la colonne sur laquelle appliquer la condition. Dans cet exemple, les deux colonnes *new-city* et *new_id* portent des conditions. Le but est d'extraire toutes les lignes concernant les clients dont la ville correspond à "New York" et dont l'identifiant est supérieur à 2.
- Cliquez dans chaque ligne et sélectionnez, à partir de la liste déroulante **Operator**, l'opérateur à utiliser pour relier la colonne d'entrée avec sa valeur associée. Dans cet exemple, l'opérateur **Equal** est sélectionné pour la colonne *new_city* et **Greater Than** pour *new_id*.
- Cliquez dans chaque ligne et définissez, dans la colonne **Value**, la valeur des colonnes d'entrée : `New York` pour la colonne *new_city* et `2` pour *new_id* dans cet exemple. Il est possible d'utiliser une valeur fixe aussi bien qu'une valeur de contexte dans ce champ.
- Double-cliquez sur le composant **tFileOutputdelimited** pour afficher la vue **Basic settings** et configurer ses paramètres de base.

- Sélectionnez le mode **Built-In** à partir de la liste déroulante **Property Type** puis cliquez sur le bouton **[...]** à côté du champ pour parcourir vos dossiers jusqu'au fichier de sortie.
- Définissez les séparateurs de lignes (**Row separator**) et de champs (**Field separator**).
- Cochez la case **Append** pour ajouter des lignes à la fin des entrées de la base de données.
- Cochez la case **Include Header** si le fichier de sortie comprend un en-tête.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma à partir du composant précédent.
- Enregistrez votre Job puis cliquez sur **F6** pour l'exécuter.

	A	B	C	D	E
1	7;married;Harry;hnewman@comp.com;New York;H.M;55677				
2	8;married;Jerry;jnewman@comp.com;New York;J.M;55677				
3	9;married;Alice;anewman@comp.com;New York;A.M;55677				
4	10;single;Jack;jnewman@comp.com;New York;J.M;55677				
5					
6					

Seuls les clients habitant la ville de New York, et dont l'identifiant est supérieur à 2, apparaissent dans la liste du fichier de sortie stocké en local.

tMicrosoftCRMOutput

Propriétés du tMicrosoftCRMOutput

Famille de composant	Business	
Fonction	Le composant **tMicrosoftOutput** écrit dans un objet de la base de données MicrosoftCRM via le service Web adéquat.	
Objectif	Ce composant permet d'écrire des données dans une base de données MicrosoftCRM.	
Basic settings	*Microsoft Webservice URL*	Saisissez l'URL du service Web permettant de se connecter à la base de données MicrosoftCRM.
	Organizename	Renseignez le nom de l'organisme qui veut accéder à la base de données MicrosoftCRM.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Domain	Saisissez le nom de domaine du serveur qui installe MicrosoftCRM.
	Host	Adresse IP du serveur de base de données MicrosoftCRM.
	Port	Numéro du port d'écoute du serveur de la base de données MicrosoftCRM.
	Action	Vous pouvez effectuer l'une des opérations suivantes sur les données MicrosoftCRM : **Insert** : insérer les données. **Update** : mettre les données à jour. **Delete** : supprimer les données.
	Time out (seconds)	Nombre de secondes d'écoute du port avant sa fermeture.
	Entity	Sélectionnez l'entité adéquate à partir de la liste déroulante.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Usage	Ce composant est utilisé comme composant de sortie. Il nécessite un composant d'entrée.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Ecrire des données dans une base de données Microsoft CRM et attribuer des conditions aux colonnes pour extraire des lignes spécifiques, page lv*.

tMSAXInput

Propriétés du tMSAXInput

Famille de composant	Business/ Microsoft AX	
Fonction	Le composant **tMSAXInput** se connecte au serveur MicrosoftAX.	
Objectif	Ce composant permet de lire les données d'un serveur MicrosoftAX et d'effectuer des requêtes sur ces données.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier où sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur.
	Domain	Saisissez le nom de domaine sur lequel est hébergé le serveur MicrosoftAX.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. iSi vous effectuez des modifications, le schéma passera automatiquement en built-in.
	Table Name	Nom de la table à lire.
	Query	Saisissez votre requête SQL en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé en tant que composant de début. Un composant de sortie est nécessaire.	
Limitation	n/a	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tMSAXOutput

Propriétés du tMSAXOutput

Famille de composant	Business/ Microsoft AX	
Fonction	Le composant **tMSAXOutput** se connecte au serveur MicrosoftAX.	
Objectif	Ce composant permet d'écrire les données d'un serveur MicrosoftAX.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier où sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur.
	Domain	Saisissez le nom de domaine sur lequel est hébergé le serveur MicrosoftAX.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur.
	Table Name	Nom de la table à écrire.
	Action on data	Vous pouvez effectuer n'importe quelle opération sur les données d'un serveur MicrosoftAX. **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une clé de recherche sur laquelle baser les opérations Update et Delete. Il est possible de définir les colonnes qui agiront comme clé de recherche à partir du schéma, pour une utilisation de base, ou à partir des options avancées (Advanced settings) pour une utilisation optimisée de ces opérations.*
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. iSi vous effectuez des modifications, le schéma passera automatiquement en built-in.

	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
	Additional columns	Cette option vous permet d'utiliser des expressions locales afin d'effectuer des actions sur les colonnes. Par exemple, vous pouvez modifier les valeurs dans la table sélectionnée : Quand vous mettez à jour ou que vous effacez des données dans une colonne, cette option vous fournit d'autres possibilités concernant les clauses WHERE en utilisant différents opérateurs de la colonne **Operator**. **Name** : nom de la colonne du schéma devant être modifiée ou insérée comme nouvelle colonne. **Operator** : sélectionnez dans la liste l'opérateur que vous souhaitez utiliser avec la clause WHERE. Cette colonne n'est pas disponible quand vous utilisez **Insert** comme action sur les données. **Data type** : type de données. **Local expression** : type d'expression locale qui sera exécuté afin de modifier ou insérer la colonne de données correspondante, par exemple *row1.[row name]*. Ou bien appuyez sur **Ctrl + espace** et sélectionnez une variable de contexte dans la liste. **Position** : sélectionnez dans la liste **Before**, **After** ou **Replace**, suivant l'action que vous voulez exécuter dans la colonne de référence. **Reference column** : type, dans une colonne de référence, que le composant peut utiliser pour insérer/remplacer la nouvelle colonne ou la colonne modifiée.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé en tant que composant de sortie. Un composant de début est nécessaire.	
Limitation	n/a	

Scénario : Insérer des données dans une table définie sur le serveur Microsoft AX

Avant de pouvoir utiliser ce composant, vérifiez que vous avez installé et lancé le serveur Microsoft AX correctement.

Ce scénario Java décrit un Job à deux composants qui utilise le composant **tMSAXOutput** afin d'insérer une ligne dans une table définie stockée sur un serveur Microsoft AX et modifie ensuite les valeurs dans l'une des colonnes insérée.

- Glissez le composant **tFixedFlowInput** ainsi que le **tMSAXOutput** de la **Palette** dans l'espace de modélisation.
- Connectez les composants à l'aide d'un lien **Row** de type **Main**.
- Double-cliquez sur le **tFixedFlowInput** afin d'afficher la vue **Component** et de définir ses propriétés.

- Sélectionnez le mode **Built-In** dans le champ **Schema Type** et cliquez sur le bouton **[...]** du champ **Edit schema** afin d'afficher une boîte de dialogue qui vous permettra de définir le schéma d'entrée.
- Cliquez sur le bouton **Plus** vert pour ajouter des colonnes dans le schéma d'entrée, trois dans cet exemple, *name*, *city* et *street* (nom, ville et rue).
- Cliquez sur **OK** pour fermer la boîte de dialogue. Une nouvelle boîte de dialogue s'ouvre et vous demande si vous voulez propager les modifications, cliquez sur **Yes** (Oui).
- Les colonnes du schéma s'affichent automatiquement dans le tableau **Values**.
- Cliquez sur la colonne **Value** et saisissez une valeur pour chaque colonne d'entrée.
- Double-cliquez sur le **tMSAXOutput** pour ouvrir la vue **Component** et définir ses propriétés.

- Sélectionnez le mode **Built-In** dans le champ **Property Type**.
- Dans le champ **Host**, saisissez l'adresse IP du serveur MicrosoftAX
- Dans le champ **Domain**, saisissez le nom du domaine qui héberge le serveur MicrosoftAX.
- Entrez votre nom d'utilisateur et votre mot de passe de connexion au serveur dans les champs correspondants.
- Dans le champ **Table Name**, saisissez le nom de la table dans la quelle vous souhaitez écrire des données.
- Dans la liste **Action on data**, sélectionnez l'action que vous désirez exécuter, **Delete** dans cet exemple.
- Cliquez sur **Sync columns** pour retrouver le schéma du composant précédent. Dans cet exemple, le but est de retrouver les trois colonnes d'entrée : *name*, *city* et *street* (nom, ville et rue).
- Si nécessaire, cliquez sur le bouton **[...]** du champ **Edit schema** afin d'ouvrir une boîte de dialogue qui vous permettra de vérifier le schéma retourné.
- Dans le tableau **Additional columns**, cliquez sur le bouton **[+]** afin d'ajouter une ligne dans laquelle vous pouvez utiliser une expression locale pour modifier les données que vous voulez écrire dans une colonne spécifique.
- Cliquez sur la colonne **Local expression** et appuyez sur **Ctrl + espace** sur votre clavier pour ouvrir la liste des variables de contexte et sélectionnez : `StringHandling.UPCASE(row2.city)+"-"+row2.street`. Cette expression écrira le nom de la ville avec la première lettre en majuscule, suivi du nom de la rue, pour former l'adresse du *Bryant park*. La colonne address de cet exemple contiendra donc la chaîne de caractères suivante : *New York-Midtown Manhattan.*

- Sauvegardez votre Job et appuyez sur **F6** pour l'exécuter.

Le **tMSAXOutput** insère dans la table *ADDRESS* du serveur MicrosoftAX une ligne contenant les trois colonnes d'entrée *name*, *city* et *street*, en plus de la nouvelle colonne *address* qui contient à la fois le nom de la ville et le nom de la rue.

Scénario 2 : Effacer des données d'une table précise sur le serveur MicrosoftAX.

Avant de pouvoir utiliser ce composant, vérifiez que vous avez installé et lancé correctement le serveur MicrosoftAX.

Ce scénario Java décrit un Job à deux composants qui utilise le **tMSAXOutput** dans le but d'effacer toutes les lignes d'une table définie qui ne correspondent pas aux données contenues dans la colonne clé sur un serveur MicrosoftAX.

Dans cet exemple, le schéma d'entrée que vous utilisez est une colonne *address* (adresse) qui contient les données suivantes : *New York-Midtown Manhattan*. Vous allez effacer du serveur MicrosoftAX toutes les adresses qui ne sont pas identiques à celle-ci.

- Glissez les composants **tFixedFlowInput** et **tMSAXOutput** de la **Palette** dans l'espace de modélisation.
- Connectez les deux composants à l'aide d'un lien **Row** de type **Main**.
- Double-cliquez sur le **tFixedFlowInput** pour afficher sa vue **Component** et définir ses propriétés.

- Sélectionnez le mode **Built-In** dans le champ **Schema Type** et cliquez sur le bouton **[...]** du champ **Edit schema** pour ouvrir une boîte de dialogue où vous pourrez définir le schéma.
- Cliquez sur le bouton **[+]** et ajoutez les colonnes d'entrée du schéma, *address* dans cet exemple.

- Cliquez sur **OK** pour fermer la boîte de dialogue. Les colonnes du schéma s'affichent automatiquement dans la list **Values**.
- Cliquez sur la colonne **Value** et saisissez une valeur pour la colonne d'entrée.
- Double-cliquez sur le **tMSAXOutput** pour ouvrir sa vue **Component** et définir ses propriétés.

- Sélectionnez le mode **Built-In** dans le champ **Property Type**.
- Dans le champ **Host**, saisissez l'adresse IP du serveur MicrosoftAX.
- Dans le champ **Domain**, entrez le nom du domaine qui héberge le serveur MicrosoftAX.
- Saisissez votre nom d'utilisateur et votre mot de passe de connexion au serveur dans les champs correspondants.
- Dans le champ **Table Name,** saisissez le nom de la table dans laquelle vous voulez écrire des données.
- Dans la liste **Action on data**, sélectionnez l'action que vos souhaitez exécuter, **Delete** dans cet exemple.
- Cliquez sur **Sync columns** pour retourner le schéma du composant précédent. Dans cet exemple, l'objectif est de récupérer la colonne d'entrée *zipcode* (code postal).
- Cliquez sur le bouton **[...]** du champ **Edit Schema** pour ouvrir la boîte de dialogue dans laquelle vous pouvez vérifier le schéma récupéré.

- Dans le schéma de sortie, cochez la case **Key** (clé) à côté du nom de la colonne.

  Quand vous sélectionnez une option différente de **Insert**, vous devez toujours définir la **Reference column** (colonne de référence) comme colonne clé.

- Cliquez sur OK pour valider les changements et fermer la boîte de dialogue.
- Dans le tableau **Additional columns**, cliquez sur le bouton **[+]** pour ajouter une ligne et définir les paramètres que le composant utilisera comme base pour l'opération d'effacement.
- Saisissez un nom (**name**), un opérateur (**operator**), un type de données (**data type**), une expression locale (**local expression**), une position (**position**) et une colonne de référence (**reference column**) dans les colonnes correspondantes à la ligne que vous venez d'ajouter. Dans cet exemple, l'objectif est d'effacer de la table *ADDRESS* du serveur MicrosoftAX toutes les lignes dans lesquelles la colonne *Address* n'est pas identique à l'adresse dans la colonne clé *address* : `New York-Midtown Manhattan.`

  Quand vous sélectionnez une action autre que **Insert** ,dans le champ **Action on data** vous devez toujours sélectionner **Replace** dans la colonne **Position**.

- Sauvegardez votre Job et appuyez sur **F6** pour l'exécuter.

Le **tMSAXOutput** supprime de la table *ADDRESS* du serveur MicrosoftAX toutes les lignes pour lesquelles les données de la colonne *address* ne sont pas identiques à celles de la colonne clé.

tOpenbravoERPInput

Propriétés du tOpenbravoERPInput

Famille de composant	Business	
Fonction	Le composant **tOpenbravoERPInput** se connecte à une entité de la base de données OpenbravoERP via le service Web adéquat.	
Objectif	Ce composant permet d'extraire les données d'une base OpenBravoERP selon certaines conditions définies dans des colonnes spécifiques.	
Basic settings	*Openbravo REST WebService URL*	Saisissez l'URL du service Web permettant de se connecter à la base de données OpenbravoERP.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Entity	Sélectionnez l'entité adéquate à partir de la liste déroulante.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Si vous effectuez des modifications, le schéma passera automatiquement en built-in. Pour ce composant, le schéma correspond à l'entité sélectionnée.
	WHERE Clause	Saisissez votre clause WHERE.
	Order by	Cochez cette case afin de sélectionner comment ordonner les résultats (les éléments de la liste déroulante dépendent de l'entité sélectionnée). **Sort** : Dans cette liste, vous pouvez choisir de trier vos résultats de manière ascendante (**Ascending**) ou descendante (**Descending**).
	First result	Saisissez le numéro de la ligne que vous souhaitez récupérer en premier.
	Max result	Saisissez le nombre maximal de résultats que vous souhaitez récupérer.
Advanced settings	*Advanced separator (for numbers)*	Cochez cette option pour modifier les séparateurs utilisés pour les nombres : **Thousands separator** : définissez le séparateur utilisé pour les milliers. **Decimal separator** : définissez le séparateur utilisé pour les décimaux.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé en tant que composant de début. Un composant de sortie est nécessaire.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Ecrire des données dans une base de données Microsoft CRM et attribuer des conditions aux colonnes pour extraire des lignes spécifiques, page lv*.

tOpenbravoERPOutput

Propriétés du tOpenbravoERPOutput

Famille de composant	Business	
Fonction	Le composant **tOpenbravoERPOutput** écrit dans un objet de la base de données OpenbravoERP via le service Web adéquat.	
Objectif	Ce composant permet d'écrire des données dans une base de données OpenbravoERP.	
Basic settings	*Openbravo REST Webservice URL*	Saisissez l'URL du service Web permettant de se connecter à la base de données OpenbravoERP.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Action on data	Sélectionnez dans la liste si vous souhaitez mettre à jour ou créer des données (**Update/Create**), ou si vous préférez en supprimer (**Remove**).
	Use existing data file	Cochez cette case afin de sélectionner votre fichier en parcourant votre répertoire.
	Entity	Sélectionnez l'entité adéquate à partir de la liste déroulante.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Usage	Ce composant est utilisé comme composant de sortie. Il nécessite un composant d'entrée.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Ecrire des données dans une base de données Microsoft CRM et attribuer des conditions aux colonnes pour extraire des lignes spécifiques, page lv*.

tSalesforceBulkExec

Les composants **tSalesforceOutputBulk** et **tSalesforceBulkExec** sont généralement utilisés ensemble pour générer en sortie le fichier défini et exécuter ensuite les actions souhaitées sur votre fichier, dans votre Salesforce.com. Cette exécution en deux étapes est unifiée dans le composant **tSalesforceOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le changement des données dans la base de données.

Propriétés du tSalesforceBulkExec

Famille de composant	Business	
Fonction	Le composant **tSalesforceBulkExec** exécute les actions définies sur les données de chargement de masse préparées.	
Objectif	En tant que composant dédié, le **tSalesforceBulkExec** vous permet d'optimiser les performances lors des exécutions d'opérations définies dans votre Salesforce.com.	
Basic settings	*Use an existing connection*	Cochez cette case pour utiliser une connexion établie à partir du **tSalesforceConnection**. Une fois cochée, la liste **Component List** apparaît, et vous permet de sélectionner le composant **tSalesforceConnection** devant être utilisé. Pour plus d'informations, consultez la section *Propriétés du tSalesforceConnection, page lxxvii*. Lorsqu'un Job contient le Job parent et le Job enfant, la liste **Component list** présente uniquement les composants de connexion du même niveau de Job. Si vous souhaitez utiliser une connexion existant dans un autre niveau de Job, vous pouvez utiliser l'onglet **Dynamic settings**, afin de partager la connexion en question. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct. Pour plus d'information concernant l'onglet **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Salesforce Webservice URL	Saisissez l'URL du service Web permettant de se connecter à la base de données Salesforce.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Bulk file path	Répertoire où sont stockées les données de masse que vous devez traiter.
	Action	Vous pouvez effectuer les opérations suivantes sur les données de l'objet Salesforce : **Insert** : insérer des données. **Update** : mettre à jour des données. **Upsert** : mettre à jour et insérer des données.

	Module	Sélectionnez l'objet adéquat dans la liste. **Custom Object** : cette option fait apparaître le champ **Costum Module Name** dans lequel vous pouvez saisir l'objet auquel vous souhaitez vous connecter.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
Advanced settings	*Rows to commit*	Saisissez le nombre de lignes devant être traitées par lot de données.
	Bytes to commit	Saisissez le nombre d'octets par lot de données devant être traités.
	Use Socks Proxy	Cochez cette case si vous souhaitez utiliser un proxy.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé comme composant de sortie. Il nécessite un composant d'entrée.	
Limitation	Les données de masse à traiter doivent être au format *.csv*.	

Scénario associé

Pour un scénario associé, consultez *Scénario : Insérer des données de masse transformées dans votre Salesforce.com, page xcvi*.

tSalesforceConnection

Propriétés du tSalesforceConnection

Famille de composant	Business	
Fonction	Le composant **tSalesforceConnection** ouvre une connexion vers un système Salesforce afin d'effectuer une transaction.	
Objectif	Ce composant permet de vous connecter à Salesforce.	
Basic settings	*Salesforce Webservice URL*	Saisissez l'URL du service Web permettant de se connecter à la base de données Salesforce.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Use Soap Compression	Cochez cette case pour activer la compression SOAP. La compression des messages SOAP permet un gain de performance.
	Use Socks Proxy	Cochez cette case si vous souhaitez utiliser un proxy.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec les composants Salesforce.	
Limitation	n/a	

Scénario associé

Pour plus d'informations relatives au fonctionnement du composant **tSalesforceConnection**, consultez *tMysqlConnection, page 471*.

tSalesforceGetDeleted

Propriétés du tSalesforceGetDeleted

Famille de composant	Business	
Fonction	Le composant **tSalesforceGetDeleted** récupère les données supprimées d'un objet Salesforce sur une période donnée.	
Objectif	Ce composant récupère les données supprimées de l'objet Salesforce spécifié sur une période spécifiée.	
Basic settings	*Use an existing connection*	Cochez cette case pour utiliser une connexion établie à partir du **tSalesforceConnection**. Une fois cochée, la liste **Component List** apparaît, et vous permet de sélectionner le composant **tSalesforceConnection** devant être utilisé. Pour plus d'informations, consultez la section *Propriétés du tSalesforceConnection, page lxxvii*. Lorsqu'un Job contient le Job parent et le Job enfant, la liste **Component list** présente uniquement les composants de connexion du même niveau de Job. Si vous souhaitez utiliser une connexion existant dans un autre niveau de Job, vous pouvez utiliser l'onglet **Dynamic settings**, afin de partager la connexion en question. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct. Pour plus d'information concernant l'onglet **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Salesforce Webservice URL	Saisissez l'URL du service Web permettant de se connecter à la base de données Salesforce.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Timeout (milliseconds)	Saisissez le délai avant suspension de la requête sur Salesforce.com.
	Module	Sélectionnez l'objet adéquat dans la liste. **Custom Object** : cette option fait apparaître le champ **Costum Module Name** dans lequel vous pouvez saisir l'objet auquel vous souhaitez vous connecter.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.

	Start Date	Saisissez entre guillemet la date à laquelle vous voulez commencer la recherche. Pour cela, utilisez le format de date suivant : “yyy-MM-dd HH:mm:ss”. Vous ne pouvez effectuer la recherche que sur les 30 derniers jours.
	End Date	Saisissez entre guillemet la date à laquelle vous voulez terminer la recherche. Pour cela, utilisez le format de date suivant : “yyy-MM-dd HH:mm:ss”.
Advanced settings	*Use Soap Compression*	Cochez cette case pour activer la compression SOAP. La compression des messages SOAP permet un gain de performance.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé comme composant de sortie. Il nécessite un composant d’entrée.	
Limitation	n/a	

Scénario : Récupérer les données supprimées du serveur Salesforce

Ce scénario décrit un Job à deux composants permettant de récupérer les données supprimées du serveur Salesforce, dans les 5 derniers jours.

- Cliquez et déposez les composants **tSalesforceGetDeleted** et **tLogRow** de la Palette dans le Job Designer.
- Reliez-les via un lien de type **Row** > **Main**.
- Double-cliquez sur le composant **tSalesforceGetDeleted** pour afficher l’onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés :

- Dans le champ **Salesforce WebService URL**, laissez l'URL du service Web Salesforce par défaut ou saisissez l'URL à laquelle vous souhaitez accéder.
- Dans les champs **Username** et **Password**, saisissez votre identifiant et votre mot de passe de connexion au service Web.
- Dans la liste **Module**, sélectionnez l'objet auquel vous souhaitez accéder. Dans ce scénario, l'objectif est d'accéder à l'objet **Account**.
- Dans la liste **Schema**, sélectionnez **Repository** puis cliquez sur le bouton **[...]** pour sélectionner le schéma du repository que vous souhaitez utiliser pour le composant. Si vous n'avez pas enregistré le schéma dans les métadonnées du repository, sélectionnez **Built-in** dans la liste **Schema** et cliquez sur le bouton **[...]** à coté du champ **Edit schema** et paramétrer le schéma manuellement.
- Dans les champs **Start Date** et **End Date**, saisissez la date de début et de fin de recherche des données supprimées en utilisant le format de date suivant : "*yyyy-MM-dd HH:mm:ss*". Vous ne pouvez effectuer la recherche que sur les 30 derniers jours. Dans ce scénario, l'objectif est de chercher les données supprimées dans les cinq derniers jours.
- Double-cliquez sur le composant **tLogRow** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés :
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent.
- Dans la zone **Mode**, sélectionnez l'option **Vertical** pour faire apparaître le résultat dans sous forme de tableau dans la console.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job tSalesforceGetDeleted_scenario at 17:17 25/06/2009.

.-------------------------------------------------------.
|                    #1. tLogRow_1                      |
+----------------------+-----------------------------+
| key                  | value                       |
+----------------------+-----------------------------+
| Id                   | 001700000003smNAAR          |
| IsDeleted            | true                        |
| MasterRecordId       | null                        |
| Name                 | sForce                      |
| Type                 | null                        |
| ParentId             | null                        |
| BillingStreet        | The Landmark at One Market  |
| BillingCity          | San Francisco               |
| BillingState         | CA                          |
| BillingPostalCode    | 94087                       |
| BillingCountry       | US                          |
| ShippingStreet       | null                        |
| ShippingCity         | null                        |
| ShippingState        | null                        |
| ShippingPostalCode   | null                        |
| ShippingCountry      | null                        |
```

La console affiche les données supprimées récupérées par le composant **tSalesforceGetDeleted** sous forme de tableau.

tSalesforceGetServerTimestamp

Propriétés du tSalesforceGetServerTimestamp

Famille de composant	Business	
Fonction	Le composant **tSalesforceGetServerTimestamp** récupère la date actuelle du serveur Salesforce.	
Objectif	Ce composant récupère la date actuelle sous forme de timestamp du serveur Salesforce.	
Basic settings	*Use an existing connection*	Cochez cette case pour utiliser une connexion établie à partir du **tSalesforceConnection**. Une fois cochée, la liste **Component List** apparaît, et vous permet de sélectionner le composant **tSalesforceConnection** devant être utilisé. Pour plus d'informations, consultez la section *Propriétés du tSalesforceConnection, page lxxvii*. Lorsqu'un Job contient le Job parent et le Job enfant, la liste **Component list** présente uniquement les composants de connexion du même niveau de Job. Si vous souhaitez utiliser une connexion existant dans un autre niveau de Job, vous pouvez utiliser l'onglet **Dynamic settings**, afin de partager la connexion en question. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct. Pour plus d'information concernant l'onglet **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Salesforce Webservice URL	Saisissez l'URL du service Web permettant de se connecter à la base de données Salesforce.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Timeout (milliseconds)	Saisissez le délai avant suspension de la requête sur Salesforce.com.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
Advanced settings	*Use Socks Proxy*	Cochez cette case si vous souhaitez utiliser un proxy.
	Use Soap Compression	Cochez cette case pour activer la compression SOAP. La compression des messages SOAP permet un gain de performance.

	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé comme composant de sortie. Il nécessite un composant d'entrée.	
Limitation	n/a	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tSalesforceGetUpdated

Propriétés du tSalesforceGetUpdated

Famille de composant	Business	
Fonction	Le composant **tSalesforceGetUpdated** récupère les données mises à jour d'un objet Salesforce sur une période donnée.	
Objectif	Ce composant récupère les données mises à jour de l'objet Salesforce spécifié sur une période spécifiée.	
Basic settings	*Use an existing connection*	Cochez cette case pour utiliser une connexion établie à partir du **tSalesforceConnection**. Une fois cochée, la liste **Component List** apparaît, et vous permet de sélectionner le composant **tSalesforceConnection** devant être utilisé. Pour plus d'informations, consultez la section *Propriétés du tSalesforceConnection, page lxxvii*. Lorsqu'un Job contient le Job parent et le Job enfant, la liste **Component list** présente uniquement les composants de connexion du même niveau de Job. Si vous souhaitez utiliser une connexion existant dans un autre niveau de Job, vous pouvez utiliser l'onglet **Dynamic settings**, afin de partager la connexion en question. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct. Pour plus d'information concernant l'onglet **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Salesforce Webservice URL	Saisissez l'URL du service Web permettant de se connecter à la base de données Salesforce.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Timeout (milliseconds)	Saisissez le délai avant suspension de la requête sur Salesforce.com.
	Module	Sélectionnez l'objet adéquat dans la liste. **Custom Object** : cette option fait apparaître le champ **Costum Module Name** dans lequel vous pouvez saisir l'objet auquel vous souhaitez vous connecter.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.

	Start Date	Saisissez entre guillemet la date à laquelle vous voulez commencer la recherche. Pour cela, utilisez le modèle de date suivant : “yyy-MM-dd HH:mm:ss”. Vous ne pouvez effectuer la recherche que sur les 30 derniers jours.
	End Date	Saisissez entre guillemets la date à laquelle vous voulez terminer la recherche. Pour cela, utilisez le modèle de date suivant : “yyy-MM-dd HH:mm:ss”.
Advanced settings	*Use Soap Compression*	Cochez cette case pour activer la compression SOAP. La compression des messages SOAP permet un gain de performance.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé comme composant de sortie. Il nécessite un composant d’entrée.	
Limitation	n/a	

Scénario associé

Aucun scénario n’est disponible pour ce composant.

tSalesforceInput

Propriétés du tSalesforceInput

Famille de composant	Business	
Fonction	Le composant **tSalesforceInput** se connecte à un objet de la base de données Salesforce via le service Web adéquat.	
Objectif	Ce composant permet d'extraire les données d'une base de données Salesforce à l'aide d'une requête.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier où sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant connexion de base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Use an existing connection	Cochez cette case pour utiliser une connexion établie à partir du **tSalesforceConnection**. Une fois cochée, la liste **Component List** apparaît, et vous permet de sélectionner le composant **tSalesforceConnection** devant être utilisé. Pour plus d'informations, consultez la section *Propriétés du tSalesforceConnection, page lxxvii*. Lorsqu'un Job contient le Job parent et le Job enfant, la liste **Component list** présente uniquement les composants de connexion du même niveau de Job. Si vous souhaitez utiliser une connexion existant dans un autre niveau de Job, vous pouvez utiliser l'onglet **Dynamic settings**, afin de partager la connexion en question. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct. Pour plus d'information concernant l'onglet **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Salesforce Webservice URL	Saisissez l'URL du service Web permettant de se connecter à la base de données Salesforce.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.

	Timeout (milliseconds)	Saisissez le délai avant suspension de la requête sur Salesforce.com.
	Module	Sélectionnez l'objet adéquat dans la liste. **Use Custom Object** : cette option fait apparaître le champ **Costum Module Name** dans lequel vous pouvez saisir l'objet auquel vous souhaitez vous connecter.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champ qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.. Pour ce composant, le schéma dépend du **Module** sélectionné. *Pour retrouver une colonne d'un module lié, il est nécessaire de définir la colonne, d'une manière précise, dans la vue Edit Schema, sinon la requête relationnelle ne fonctionnera pas. La syntaxe correcte est :`NameofCurrentModule_NameofLinkedModule_NameofColumnofInteresst`.*
	Query condition	Saisissez la requête permettant de sélectionner les données à extraire. Exemple : `account_name='Talend'`
	Manual input of SOQL query	Cochez cette case pour saisir manuellement la requête Salesforce que vous souhaitez effectuer dans le champ **Query**.
Advanced settings	*Batch Size*	Nombre d'enregistrements dans chaque lot récupéré.
	Use Socks Proxy	Cochez cette case si vous souhaitez utiliser un proxy.
	Normalize delimited (for child relationship)	Caractère, chaîne ou expression régulière utilisé pour normaliser les données collectées à l'aide de requêtes effectuées sur les relations hiérarchiques entre les différents objets Salesforce.
	Column name delimiter (for child relationship)	Caractère, chaîne ou expression régulière utilisé pour séparer le nom de l'objet parent du nom de l'objet fils lorsque vous effectuez une requête sur les relations hiérarchiques entre les différents objets Salesforce.
	Use Soap Compression	Cochez cette case pour activer la compression SOAP. La compression des messages SOAP permet un gain de performance, notamment lors des opérations de batch.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé en tant que composant de début. Un composant de sortie est nécessaire.	
Limitation	n/a	

Scénario : Utiliser des requêtes pour extraire des données d'une base de données Salesforce

Ce scénario décrit un Job comprenant quatre composants, utilisés pour extraire des ensembles spécifiques de données d'objets parents et enfants dans une base de données Salesforce.

- Déposez deux composants **tSalesforceInput** et deux **tLogRow** dans l'espace de modélisation graphique.

- Reliez chaque composant **tSalesforceInput** à un **tLogRow**, à l'aide de liens **Row** > **Main**.
- Reliez le **tSalesforceInput_1** au **tSalesforceInput_2** en utilisant un lien **OnSubjobOk**.
- Double-cliquez sur le **tSalesforceInput_1** afin de définir ses **Basic Settings** dans la vue **Component**.

- Dans la liste **Property Type**, sélectionnez **Built-In**.
- Dans le champ **Salesforce WebService URL**, saisissez l'URL de la base de données à laquelle vous souhaitez vous connecter.
- Renseignez vos informations d'authentification dans les champs correspondants, **Username** pour l'identifiant et **Password** pour le mot de passe.
- Saisissez la limite de temps que vous souhaitez appliquer à la requête, dans le champ **Timeout (milliseconds)**.

- Sélectionnez le **Module** (objet Salesforce) que vous souhaitez interroger.
- Cochez la case **Manual input of SOQL Query** afin de saisir votre requête dans le champ **Query**.
- Saisissez votre requête, ou votre requête relationnelle, en respectant la syntaxe SOQL requise.

Dans cet exemple, les colonnes *IsWon* et *FiscalYear* dans la requête sont situées dans le module **Opportunity** spécifié. La colonne *Name* est dans un module lié, appelé *Account*. Pour retourner une colonne d'un module lié, la syntaxe correcte est de saisir le nom du module lié, suivi par le caractère point (.), puis par le nom de la colonne appropriée. Ici, la requête requise est donc : `"SELECT IsWon, FiscalYear, Account.Name FROM Opportunity"`.

Pour retrouver une colonne d'un modèle lié, il est nécessaire de définir la colonne d'une manière précise dans la vue Edit Schema. La syntaxe correcte est :NameofCurrentModule_NameofLinkedModule_NameofColumnofInterest. Dans notre exemple, la colonne doit se nommer : Opportunity_Account_Name. Si cette syntaxe n'est pas respectée, les données de la table liée ne seront pas retournées.

- Sélectionnez **Built-In** dans la liste **Schema** puis cliquez sur le bouton **[...]** à côté du champ **Edit schema**, afin d'ouvrir l'éditeur du schéma.

- Modifiez le schéma comme nécessaire, en utilisant les boutons **[+]** et **[x]**.
- Ajoutez une nouvelle colonne pour les champs extraits de la colonne *Name* du module *Account*.
- Appelez cette colonne *Opportunity_Account_Name*.
- Cliquez sur **OK** afin de sauvegarder les modifications et fermer l'éditeur du schéma.
- Double-cliquez sur le **tSalesforceInput_2** afin de définir ses **Basic settings** dans la vue **Component**.

- Dans la liste **Property Type**, sélectionnez **Built-In**.
- Dans le champ **Salesforce WebService URL**, saisissez l'URL de la base de données à laquelle vous souhaitez vous connecter.
- Saisissez vos informations d'authentification dans les champs **Username** pour l'identifiant et **Password** pour le mot de passe.
- Saisissez la limite de temps que vous souhaitez appliquer sur la requête, dans le champ **Timeout (milliseconds)**.
- Sélectionnez le **Module** (objet Salesforce) que vous souhaitez interroger.
- Cochez la case **Manual input of SOQL Query** pour saisir votre requête dans le champ **Query**.
- Saisissez votre requête ou votre requête relationnelle, en respectant la syntaxe SOQL requise.

Dans cet exemple, l'objectif est d'extraire les champs *Id* et *CaseNumber* du module *Case*, ainsi que les champs *Name* du module *Account*. La requête est donc : `"SELECT Id, CaseNumber, Account.Name FROM Case"`.

- Sélectionnez **Built-In** dans la liste **Schema** et cliquez sur le bouton **[...]** à côté du champ **Edit schema**, afin d'ouvrir l'éditeur du schéma.

- Modifiez le schéma comme nécessaire, en utilisant les boutons **[+]** et **[x]**.
- Ajouter une nouvelle colonne pour les champs extraits de la colonne *Name* du module *Account*.

- Nommez cette colonne *Case_Account_Name*.
- Cliquez sur **OK** pour sauvegarder les modifications et fermer l'éditeur du schéma.
- Cliquez sur chaque composant **tLogRow** afin de définir leurs **Basic settings** comme nécessaire.

Dans cet exemple, vous n'avez pas besoin de modifier les paramètres des **tLogRow**.

- Appuyez sur **F6** pour exécuter le Job.
  Les résultats sont affichés dans la console de la vue **Run** :

```
|true|2007|GenePoint
|true|2007|Burlington Textiles Corp of America
|true|2007|United Oil & Gas Corp.
|false|2007|United Oil & Gas Corp.
|true|2007|United Oil & Gas Corp.
|false|2007|United Oil & Gas Corp.
|true|2007|United Oil & Gas Corp.
|true|2007|United Oil & Gas Corp.
|false|2007|GenePoint
|true|2007|GenePoint
|true|2007|Edge Communications
|false|2007|Edge Communications
|true|2007|Edge Communications
|false|2007|Pyramid Construction Inc.
|false|2007|Dickenson plc
|true|2007|Grand Hotels & Resorts Ltd
|true|2007|Grand Hotels & Resorts Ltd
|false|2007|Grand Hotels & Resorts Ltd
50070000005DBELAA4|00001000|Edge Communications
50070000005DBEMAA4|00001001|United Oil & Gas Corp.
50070000005DBENAA4|00001002|United Oil & Gas Corp.
50070000005DBEOAA4|00001003|Express Logistics and Transport
50070000005DBEPAA4|00001004|Express Logistics and Transport
50070000005DBEQAA4|00001005|Express Logistics and Transport
50070000005DBERAA4|00001006|GenePoint
50070000005DBESAA4|00001007|Grand Hotels & Resorts Ltd
50070000005DBETAA4|00001008|Grand Hotels & Resorts Ltd
50070000005DBEUAA4|00001009|United Oil & Gas, UK
50070000005DBEVAA4|00001010|United Oil & Gas, Singapore
```

tSalesforceOutput

Propriétés du tSalesforceOutput

Famille de composant	Business	
Fonction	Le composant **tSalesforceOutput** écrit dans un objet de la base de données Salesforce via le service Web adéquat.	
Objectif	Ce composant permet d'écrire des données dans une base de données Salesforce.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier où sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant connexion de base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Use an existing connection	Cochez cette case pour utiliser une connexion établie à partir du **tSalesforceConnection**. Une fois cochée, la liste **Component List** apparaît, et vous permet de sélectionner le composant **tSalesforceConnection** devant être utilisé. Pour plus d'informations, consultez la section *Propriétés du tSalesforceConnection, page lxxvii*. Lorsqu'un Job contient le Job parent et le Job enfant, la liste **Component list** présente uniquement les composants de connexion du même niveau de Job. Si vous souhaitez utiliser une connexion existant dans un autre niveau de Job, vous pouvez utiliser l'onglet **Dynamic settings**, afin de partager la connexion en question. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct. Pour plus d'information concernant l'onglet **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Salesforce Webservice URL	Saisissez l'URL du service Web permettant de se connecter à la base de données Salesforce.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.

	Timeout (milliseconds)	Saisissez le délai avant suspension de la requête sur Salesforce.com.
	Action	Vous pouvez effectuer l'une des opérations suivantes sur les données de l'objet Salesforce : **Insert** : insérer les données. **Update** : mettre les données à jour. **Delete** : supprimer les données. **Upsert** : mettre à jour et insérer les données.
	Module	Sélectionnez l'objet adéquat dans la liste. **Custom Object** : cette option fait apparaître le champ **Costum Module Name** dans lequel vous pouvez saisir l'objet auquel vous souhaitez vous connecter.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
Advanced settings	*Extended Output*	Cette case est cochée par défaut. Elle permet de transférer les données de sortie en lot. Dans le champ **Rows to commit**, spécifiez le nombre de lignes par lot.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
	Error logging file	Renseignez ce champ si vous souhaitez créer un fichier contenant les logs d'erreur. Pour cela, cliquez sur le bouton [...] à coté de ce champ pour définir le chemin d'accès et le nom du fichier
	Use Socks Proxy	Cochez cette case si vous souhaitez utiliser un proxy.
	Use Soap Compression	Cochez cette case pour activer la compression SOAP. La compression des messages SOAP permet un gain de performance.
	Retrieve inserted ID	Cochez cette case afin de permettre à Salesforce.com de retourner l'ID Salesforce produit pour une nouvelle ligne devant être insérée. La colonne ID est ajoutée au schéma des données traitées dans Salesforce.com. Cette option est disponible uniquement si vous avez sélectionné l'action **Insert**, mais non en mode batch, c'est-à-dire sans l'option **Extended Output**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant est utilisé comme composant de sortie. Il nécessite un composant d'entrée.
Limitation	n/a

Scénario : Supprimer des données de l'objet Account

Ce scénario décrit un Job à deux composants permettant de supprimer une entrée de l'objet Account.

- Cliquez et déposez les composants **tSalesforceInput** et **tSalesforceOutput** de la **Palette** dans le Job Designer.
- Reliez-les via un lien de type **Row** > **Main**.
- Double-cliquez sur le composant **tSalesforceInput** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés :

- Dans la liste **Property Type**, sélectionnez **Repository** si vous avez stocké la connexion au serveur Salesforce dans les métadonnées du Repository. Les autres champs seront renseignés automatiquement. Si vous n'avez pas stocké la connexion dans les métadonnées, sélectionnez **Built-in** dans la liste et renseignez les champs manuellement.
- Dans le champ **Salesforce WebService URL**, laissez l'URL du service Web Salesforce par défaut ou saisissez l'URL à laquelle vous souhaitez accéder.
- Dans les champs **Username** et **Password**, saisissez votre identifiant et votre mot de passe de connexion au service Web.
- Dans la liste **Module**, sélectionnez l'objet auquel vous souhaitez accéder. Dans ce scénario, l'objectif est d'accéder à l'objet **Account**.

- Dans la liste **Schema**, sélectionnez **Repository** puis cliquez sur le bouton **[...]** pour sélectionner le schéma du repository que vous souhaitez utiliser pour le composant. Si vous n'avez pas enregistré le schéma dans les métadonnées du repository, sélectionnez **Built-in** dans la liste **Schema** et cliquez sur le bouton **[...]** à coté du champ **Edit schema** et paramétrer le schéma manuellement.
- Dans le champ **Query Condition**, saisissez la requête que vous souhaitez appliquer. Pour ce scénario, l'objectif est de récupérer le ou les clients dont le nom est *sForce*., ainsi, saisissez la requête : `"name='sForce'"`.
- Pour une requête plus avancée, cochez la case **Manual input of SOQL query** et saisissez-la manuellement.
- Double-cliquez sur le composant **tSalesforceOutput** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés :

- Dans le champ **Salesforce WebService URL**, laissez l'URL du service Web Salesforce par défaut ou saisissez l'URL à laquelle vous souhaitez accéder.
- Dans les champs **Username** et **Password**, saisissez votre identifiant et votre mot de passe de connexion au service Web.
- Dans la liste **Action**, sélectionnez l'action que vous souhaitez effectuer. Ici, sélectionnez **Delete** pour supprimer le compte *sForce* sélectionné dans le composant précédent.
- Dans la liste **Module**, sélectionnez l'objet auquel vous souhaitez accéder. Dans ce scénario, l'objectif est d'accéder à l'objet **Account**.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent.
- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

Consultez le contenu de l'objet **Account** et vérifiez que le compte *sForce* a été supprimé du serveur.

tSalesforceOutputBulk

Les composants **tSalesforceOutputBulk** et **tSalesforceBulkExec** sont généralement utilisés ensemble pour générer en sortie le fichier défini et exécuter ensuite les actions souhaitées sur votre fichier, dans votre Salesforce.com. Cette exécution en deux étapes est unifiée dans le composant **tSalesforceOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le changement des données dans la base de données.

Propriétés du tSalesforceOutputBulk

Famille de composant	Business	
Fonction	Le composant **tSalesforceOutputBulk** génère des fichiers dans un format approprié au traitement de masse.	
Objectif	Ce composant prépare le fichier devant être traité par le **tSalesForceBulkExec** afin d'exécuter des actions sur Salesforce.com.	
Basic settings	*File Name*	Parcourez le répertoire où stocker les fichiers générés.
	Append	Cochez cette case afin d'écrire à la suite des données existantes. Sinon, les données seront écrasées.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé avec le **tSalesforceBulkExec**. Ils permettent d'optimiser les performances d'alimentation ou de modification des informations sur Salesforce.com.	
Limitation	n/a	

Scénario : Insérer des données de masse transformées dans votre Salesforce.com

Ce scénario décrit un Job à six composants qui transforme des données *.csv* appropriées à un traitement de masse, les charge dans Salesforce.com et affiche ensuite les résultats de l'exécution du Job dans la console.

Ce Job est réalisé en deux étapes : préparer les données en les transformant, puis traiter les données transformées.

Avant de commencer ce scénario, vous devez préparer le fichier original comprenant les données devant être traitées par le Job. Dans cet exemple d'utilisation, le fichier est *sforcebulk.txt*, il contient des informations relatives aux clients.

Afin de créer et exécuter ce Job, procédez comme suit :

- Glissez-déposez de la **Palette** dans l'espace de modélisation graphique les composants **tFileInputDelimited**, **tMap**, **tSalesforceOutputBulk**, **tSalesforceBulkExec** et deux **tLogRow**.
- Utilisez un lien **Row** > **Main** afin de connecter le **tFileInputDelimited** au **tMap**, et le lien **Row** > **out1** du **tMap** au **tSalesforceOutputBulk**.
- Utilisez un lien **Row** > **Main** et un lien **Row** > **Reject** pour connecter le **tSalesforceBulkExec** aux deux **tLogRow** respectivement.
- Utilisez un lien **Trigger** > **OnSubjobOk** afin de connecter le composant **tFileInputDelimited** au **tSalesforceBulkExec**.
- Double-cliquez sur le **tFileInputDelimited** afin d'afficher sa vue **Component** et définir ses propriétés.

- Dans la liste **Property Type**, sélectionnez **Repository** si vous avez déjà stocké votre connexion Salesforce sous le noeud **Metadata** dans la vue **Repository**. Les propriétés sont automatiquement remplies. Si vous n'avez pas défini le serveur de connexion localement dans le Repository, renseignez les informations manuellement, après avoir sélectionné **Built-in** dans la liste **Property Type**.
  Pour plus d'informations à propos des métadonnées, consultez le chapitre *Gestion des métadonnées*.

- A côté du champ **File name/Stream**, cliquez sur le bouton afin de parcourir votre répertoire jusqu'à votre fichier, dans cet exemple *sforcebulk.txt*.

- Dans la liste **Schema**, sélectionnez **Repository**, puis cliquez sur le bouton **[...]** afin d'ouvrir la boîte de dialogue dans laquelle vous pourrez sélectionner le schéma du Repository à utiliser pour ce composant. Si vous n'avez pas défini localement votre fichier dans les métadonnées, sélectionnez **Built-in** dans la liste **Schema**, puis cliquez sur le bouton [...]à côté du champ **Edit schema**, afin d'ouvrir une boîte de dialogue dans laquelle vous pourrez définir le schéma manuellement. Dans ce scénario, le schéma se compose de quatre colonnes : *Name*, *ParentId*, *Phone* et *Fax*.

Column	Key	Type	N..	Date Patt...	Length	Pre...	D...	Co...
Name	☐	String	☑					
ParentId	☐	String	☑					
Phone	☐	String	☑					
Fax	☐	String	☑					

- Selon votre fichier d'entrée, paramétrez les autres champs, comme **Row Separator**, **Field Separator**...

- Double-cliquez sur le composant **tMap** afin d'ouvrir son éditeur et définir la transformation.

- Glissez-déposez toutes les colonnes de la table d'entrée dans la table de sortie.

- Ajoutez `.toUpperCase()` à la fin de la colonne *Name*.
- Cliquez sur **OK** afin de valider la transformation.
- Double-cliquez sur le **tSalesforceOutputBulk** afin d'afficher sa vue **Component** et définir ses propriétés.

- Dans le champ **File Name**, parcourez votre répertoire jusqu'à l'emplacement où vous souhaitez stocker les données *.csv* générées pour un traitement de masse.
- Cliquez sur le bouton **Sync columns** afin d'importer le schéma du composant précédent.
- Double-cliquez sur le **tSalesforceBulkExect** afin d'afficher sa vue **Component** et définir ses propriétés.

- Utilisez l'URL par défaut du service Web Salesforce, ou saisissez l'URL à laquelle vous souhaitez accéder.
- Dans les champs **Username** et **Password**, saisissez votre identifiant et votre mot de passe de connexion au service Web.

- Dans le champ **Bulk file path**, parcourez votre répertoire jusqu'à l'emplacement où est stocké le fichier *.csv* généré par le **tSalesforceOutputBulk**.
- Dans la liste **Action**, sélectionnez l'action que vous voulez effectuer sur les données de masse. Dans cet exemple, sélectionnez **insert**.
- Dans la liste **Module**, sélectionnez l'élément auquel vous souhaitez accéder, **Account** dans cet exemple.
- Dans la liste **Schema**, sélectionnez **Repository** et cliquez sur le bouton **[...]** afin d'ouvrir une boîte de dialogue dans laquelle vous pourrez sélectionner le schéma du Repository à utiliser pour ce composant. Si vous n'avez pas défini votre schéma localement dans les métadonnées, sélectionnez **Built-in** dans la liste **Schema**, puis cliquez sur le bouton **[...]** à côté du champ **Edit schema**, afin d'ouvrir une boîte de dialogue dans laquelle vous pourrez définir votre schéma manuellement. Dans cet exemple, définissez-le comme le schéma précédent.
- Double-cliquez sur le **tLogRow_1** afin d'afficher sa vue **Component** et définir ses propriétés.

- Cliquez sur le bouton **Sync columns** afin de récupérer le schéma du composant précédent.
- Sélectionnez le mode **Table** afin d'afficher les résultats d'exécution dans un tableau.
- Répétez l'opération avec le **tLogRow_2**.
- Sauvegardez votre Job, et appuyez sur **F6** pour l'exécuter.

Dans la console de la vue **Run**, vous pouvez vérifier les résultats d'exécution.

```
[statistics] connected
.-----------------------------------+--------+--------------+--------------+-----------------
|                                                   tLogRow_1
|=----------------------------------+--------+--------------+--------------+-----------------
|Name                               |ParentId|Phone         |Fax           |salesforce_id
|=----------------------------------+--------+--------------+--------------+-----------------
|BURLINGTON TEXTILES CORP OF AMERICA|null    |(336) 222-7000|(336) 222-8000|0018000000gfDyDAA
|DICKENSON PLC                      |null    |(785) 241-6200|(785) 241-6201|0018000000gfDyEAA
|GENEPOINT                          |null    |(650) 867-3450|(650) 867-9895|0018000000gfDyFAA
'-----------------------------------+--------+--------------+--------------+-----------------

.-------------------------+--------+--------------+--------------+-------------------------
|                                                                   tLogRow_2
|=------------------------+--------+--------------+--------------+-------------------------
|Name                     |ParentId|Phone         |Fax           |error
|=------------------------+--------+--------------+--------------+-------------------------
|EDGE COMMUNICATIONS      |wrong id|(512) 757-6000|(512) 757-9000|MALFORMED_ID:Parent Accoun
|GRAND HOTELS & RESORTS LTD|wrong id|(312) 596-1000|(312) 596-1500|MALFORMED_ID:Parent Accoun
'-------------------------+--------+--------------+--------------+-------------------------

[statistics] disconnected
Job tSalesforcebulk ended at 18:17 13/04/2010. [exit code=0]
```

Dans le tableau du **tLogRow_1**, vous pouvez lire les données insérées dans votre Salesforce.com.

Dans le tableau du **tLogRow_2**, vous pouvez lire les données rejetées à cause d'une incompatibilité avec les éléments **Account** auxquels vous avez accédé.

Tous les noms de clients sont écrits en majuscules.

tSalesforceOutputBulkExec

Les composants **tSalesforceOutputBulk** et **tSalesforceBulkExec** sont généralement utilisés ensemble pour générer en sortie le fichier défini et exécuter ensuite les actions souhaitées sur votre fichier, dans votre Salesforce.com. Cette exécution en deux étapes est unifiée dans le composant **tSalesforceOutputBulkExec**. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le changement des données dans la base de données.

Propriétés du tSalesforceOutputBulkExec

Famille de composant	Business	
Fonction	Le composant **tSalesforceOutputBulkExec** exécute les actions définies sur les données *.csv* pour Salesforce.com.	
Objectif	En tant que composant dédié, le **tSalesforceOutpuBulkExec** permet d'optimiser les performances lors des opérations définies sur les données, dans votre Salesforce.com.	
Basic settings	*Use an existing connection*	Cochez cette case pour utiliser une connexion établie à partir du **tSalesforceConnection**. Une fois cochée, la liste **Component List** apparaît, et vous permet de sélectionner le composant **tSalesforceConnection** devant être utilisé. Pour plus d'informations, consultez la section *Propriétés du tSalesforceConnection, page lxxvii*. Lorsqu'un Job contient le Job parent et le Job enfant, la liste **Component list** présente uniquement les composants de connexion du même niveau de Job. Si vous souhaitez utiliser une connexion existant dans un autre niveau de Job, vous pouvez utiliser l'onglet **Dynamic settings**, afin de partager la connexion en question. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct. Pour plus d'information concernant l'onglet **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Salesforce Webservice URL	Saisissez l'URL du service Web permettant de se connecter à la base de données Salesforce.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Bulk file path	Chemin d'accès au répertoire où sont stockées les données de masse à traiter.
	Action	Vous pouvez effectuer l'une des opérations suivantes sur les données de l'objet Salesforce : **Insert** : insérer les données. **Update** : mettre les données à jour. **Upsert** : mettre à jour et insérer les données.

	Module	Sélectionnez l'objet adéquat dans la liste. **Custom Object** : cette option fait apparaître le champ **Costum Module Name** dans lequel vous pouvez saisir l'objet auquel vous souhaitez vous connecter.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
Advanced settings	*Rows to commit*	Saisissez le nombre de lignes devant être traitées par lot de données.
	Bytes to commit	Saisissez le nombre d'octets par lot de données devant être traités.
	Use Socks Proxy	Cochez cette case si vous souhaitez utiliser un proxy. Dans ce cas, vous devez renseigner les paramètres du proxy dans les champs qui apparaissent, une fois la case cochée.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé comme composant de sortie. Il nécessite un composant d'entrée.	
Limitation	Les données de masse à traiter dans Salesforce.com doivent être au format *.csv*	

Scénario : Insérer des données de masse dans votre Salesforce.com

Ce scénario décrit un Job à quatre composants qui charge des données en masse dans Salesforce.com, exécute les actions définies, et affiche les résultats d'exécution du Job dans la console.

Avant de réaliser ce Job, vous devez préparer votre fichier d'entrée, contenant les données devant être traitées par le Job. De cet exemple, le fichier *sforcebulk.txt*, contient des informations relatives aux clients.

Afin de créer et exécuter le Job, procédez comme suit :

- Glissez-déposez de la Palette dans l'espace de modélisation graphique un composant **tFileInputDelimited**, un **tSalesforceOutputBulkExec**, et deux **tLogRow**.
- Utilisez un lien **Row** > **Main** pour connecter le **tFileInputDelimited** au **tSalesforceOutputBulkExec**.
- Utilisez un lien **Row** > **Main** et un lien **Row** > **Reject** afin de connecter le **tSalesforceOutputBulkExec** aux deux **tLogRow** respectifs.
- Double-cliquez sur le **tFileInputDelimited** afin d'afficher sa vue **Component** et définir ses propriétés.

- Dans la liste **Property Type**, sélectionnez **Repository** si vous avez déjà stocké votre connexion au serveur Salesforce dans le noeud **Metadata** de la vue **Repository**. Les propriétés sont automatiquement renseignées. Si vous n'avez pas défini la connexion au serveur localement dans le Repository, remplissezl es champs manuellement, après avoir sélectionné **Built-in** dans la liste **Property Type**.
  Pour plus d'informations concernant les métadonnées, consultez le chapitre *Gestion des métadonnées*.
- A côté du champ **File name/Stream**, cliquez sur le bouton afin de sélectionner votre fichier d'entrée, par exemple *sforcebulk.txt*.
- Dans la liste **Schema**, sélectionnez **Repository**, puis cliquez sur le bouton **[...]** pour ouvrir une boîte de dialogue dans laquelle vous pourrez sélectionner le schéma à utiliser pour ce composant. Si vous n'avez pas défini votre schéma localement dans les métadonnées, sélectionnez **Built-in** dans la liste **Schema**, puis cliquez sur le bouton **[...]** à côté du champ **Edit schema**, afin d'ouvrir une boîte de dialogue où vous pourrez définir le schéma manuellement. dans ce scénario, le schéma se compose de quatre colonnes : *Name*, *ParentId*, *Phone* et *Fax*.

Column	Key	Type	N..	Date Patt...	Length	Pre...	D...	Co...
Name		String	✓					
ParentId		String	✓					
Phone		String	✓					
Fax		String	✓					

- Configurez les autres propriétés selon votre fichier d'entrée, comme **Row Separator**, **Field Separator**...
- Double-cliquez sur le composant **tSalesforceOutputBulkExec** afin d'afficher sa vue **Component** et définir ses propriétés.

- Dans le champ **Salesforce WebService URL**, utilisez l'URL par défaut du service Web Salesforce, ou saisissez l'URL à laquelle vous souhaitez accéder.
- Dans les champs **Username** et **Password**, saisissez votre identifiant et votre mot de passe de connexion au service Web.
- Dans le champ **Bulk file path**, parcourez votre répertoire jusqu'à l'emplacement où vous stockez les données *.csv* devant être traitées.

Le fichier de masse à traiter doit être au format *.csv*.

- Dans la liste **Action**, sélectionnez l'action à effectuer sur les données de masse, **insert** dans cet exemple.
- Dans la liste **Module**, sélectionnez l'élement auquel vous souhaitez accéder, **Account** dans cet exemple.
- Dans la liste **Schema**, sélectionnez **Repository**, puis cliquez sur le bouton **[...]** afin d'ouvrir une boîte de dialogue dans laquelle vous pourrez sélectionner le schéma du Repository à utiliser pour ce composant. Si vous n'avez pas défini votre schéma localement dans les métadonnées, sélectionnez **Built-in** dans la liste **Schema**, puis cliquez sur le bouton **[...]** à côté du champ **Edit schema** afin d'ouvrir une boîte de dialogue où vous pourrez définir manuellement le schéma. Dans cet exemple, définissez le schéma comme précédemment.
- Double-cliquez sur le **tLogRow_1** afin d'afficher sa vue **Component** et définir ses propriétés.

- Cliquez sur le bouton **Sync columns** afin de récupérer le schéma du composant précédent.
- Sélectionnez le mode **Table** afin d'afficher les résultats d'exécution dans un tableau.
- Répétez l'opération pour le **tLogRow_2**.
- Sauvegardez votre Job et appuyez sur **F6** afin de l'exécuter.

Dans la console de la vue **Run**, vous pouvez vérifier les résultats d'exécution.

```
statistics] connected
-----------------------------------+--------+---------------+---------------+----------------
                                                   tLogRow_1
-----------------------------------+--------+---------------+---------------+----------------
ame                                |ParentId|Phone          |Fax            |salesforce_id
-----------------------------------+--------+---------------+---------------+----------------
urlington Textiles Corp of America|null    |(336) 222-7000|(336) 222-8000|0018000000gfDyD
ickenson plc                       |null    |(785) 241-6200|(785) 241-6201|0018000000gfDyE
enePoint                           |null    |(650) 867-3450|(650) 867-9895|0018000000gfDyF
-----------------------------------+--------+---------------+---------------+----------------

-------------------------+--------+---------------+---------------+-------------------------
                                                                      tLogRow_2
-------------------------+--------+---------------+---------------+-------------------------
ame                      |ParentId|Phone          |Fax            |error
-------------------------+--------+---------------+---------------+-------------------------
dge Communications       |wrong id|(512) 757-6000|(512) 757-9000|MALFORMED_ID:Parent Acco
rand Hotels & Resorts Ltd|wrong id|(312) 596-1000|(312) 596-1500|MALFORMED_ID:Parent Acco
-------------------------+--------+---------------+---------------+-------------------------
```

Dans le tableau du **tLogRow_1**, vous pouvez lire les données insérées dans Salesforce.com.

Dans le tableau du **tLogRow_2**, vous pouvez lire les données rejetées à cause d'une incompatibilité avec les éléments **Account** auxquels vous avez accédé.

Si vous souhaitez transformer les données d'entrée avant de les charger, vous devez utiliser un **tSalesforceOutputBulk** et un **tSalesforceBulkExec** pour réaliser le Job complet. Pour plus d'informations à propos de l'utilisation de ces deux composants, consultez *Scénario : Insérer des données de masse transformées dans votre Salesforce.com, page xcvi*.

tSAPCommit

Propriétés du tSAPCommit

Ce composant est étroitement lié aux composants **tSAPConnection** et **tSAPRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Business/SAP	
Fonction	Valide les données traitées dans un Job, via un serveur connecté.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*SAPConnection Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tSAPConnection** dans la liste.
	Release Connection	Cette case est sélectionnée par défaut. Elle vous permet de fermer une connexion à une base de données, une fois le commit effectué; Décochez cette case afin de continuer à utiliser la connexion sélectionnée, une fois que le composan a exécuté sa tâche. *Si vous utilisez un lien* **Row** > **Main** *pour relier le* **tSAPCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Release connection**, *ou votre connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants SAP et notamment avec les composants **tSAPConnection** et **tSAPRollback**..	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tSAPConnection** et **tSAPRollback**. Ils sont généralement utilisés ensemble lors de transactions, avec un **tSAPConnection** permettant d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au **tSAPCommit**, consultez *tMysqlConnection, page 471*.

tSAPConnection

Propriétés du tSAPConnection

Famille de composant	Business	
Fonction	Le composant **tSAPConnection** ouvre une connexion vers un système SAP afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers un système SAP.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier où sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Connection configuration	**Client type** : Saisissez votre code de connexion SAP habituel. **Userid :** Saisissez l'identifiant de connexion utilisateur. **Password** : Saisissez le mot de passe. **Language** : Spécifiez la langue. **Host name** : Saisissez l'adresse IP du système SAP. **System number :** Saisissez le numéro du système.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec les composants SAP.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez les *Scénario 1 : Récupérer les métadonnées d'un système SAP, page cxi*.

tSAPInput

Propriétés du tSAPInput

Famille de composant	Business	
Fonction	Le composant **tSAPInput** se connecte à un système SAP via l'adresse IP du système.	
Objectif	Il permet d'extraire des données d'un système SAP à n'importe quel niveau en faisant appel à des fonctions RFC ou BAPI.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier où sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant connexion de base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Use an existing connection	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient le Job parent et le Job enfant, la liste **Component list** présente uniquement les composants de connexion du même niveau de Job. Si vous souhaitez utiliser une connexion existant dans un autre niveau de Job, vous pouvez utiliser l'onglet **Dynamic settings**, afin de partager la connexion en question. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct. Pour plus d'information concernant l'onglet **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Connection configuration	**Client** : Saisissez votre code de connexion SAP habituel. **Userid :** Saisissez l'identifiant de connexion utilisateur. **Password** : Saisissez le mot de passe. **Language** : Spécifiez la langue. **Host name** : Saisissez l'adresse IP du système SAP. **System number :** Saisissez le numéro du système.

	FunName	Saisissez le nom de la fonction que vous souhaitez utiliser pour récupérer les données.
	Initialize input	Configurez les paramètres d'entrée. **Parameter Value** : Saisissez entre guillemets les valeurs correspondant au paramètre défini dans la colonne **Parameter Name**. **Type** : Sélectionnez le type d'entrée à récupérer. **Table Name (Structure Name)** : Saisissez entre guillemets le nom de la table. **Parameter Name** : Saisissez entre guillemets le nom du champ correspondant à la table définie dans la colonne **Table Name**. Lorsque plusieurs valeurs de paramètres utilisent le même nom de paramètre, vous devez saisir ces valeurs dans une seule ligne et les séparer par des virgules.
	Outputs	Configurez les paramètres des schémas à transmettre en sortie. **Schema** : Saisissez le nom du schéma à transmettre en sortie. **Type (for iterate)** : Sélectionnez le type de sortie que vous souhaitez obtenir. **Table Name (Structure Name)** : Saisissez entre guillemets le nom de la table. **Mapping** : Saisissez entre guillemets le nom des champs dont vous souhaitez récupérer les données. Vous pouvez définir autant de liens de sortie **Main** pour écrire des données que de schémas ajoutés au tableau **Outputs**. Ainsi, les données peuvent être groupées dans différents fichiers.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : RunIf, OnComponentOk; OnComponentError, OnSubjobOk, OnSubjobError. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate **Trigger** : Runif, OnComponentOk, OnComponentError, OnSubjobOk, OnSubjobError. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur Talend Open Studio**.
Advanced settings	*Release Connection*	Décochez cette case afin de continuer à utiliser la connexion sélectionnée, une fois que le composant a exécuté sa tâche.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant est généralement utilisé en tant que composant de début et requiert un composant de sortie.
Limitation	n/a

Scénario 1 : Récupérer les métadonnées d'un système SAP

Les composants SAP (**tSAPInput** et **tSAPOutput**) ainsi que l'assistant SAP sont basés sur des bibliothèques validées et fournies par SAP (JCO) permettant à l'utilisateur d'appeler les fonctions RFC ou BAPI et de récupérer les données du système SAP au niveau des tables, RFC ou BAPI.

Ce scénario utilise l'assistant SAP guidant l'utilisateur à travers les différentes étapes de création de connexions SAP et des fonctions RFC et BAPI. L'assistant SAP est uniquement disponible pour les utilisateurs **Talend Integration Suite**. Si vous êtes utilisateur de **Talend Open Studio** ou **Talend On Demand**, vous devez configurer les propriétés du composant **tSAPInput** manuellement.

Ce scénario Java utilise un assistant SAP pour créer une connexion au système SAP, appelle une fonction BAPI pour récupérer les informations d'une entreprise stockées dans le système SAP et les affiche dans **Talend Open Studio**.

Les figures ci-dessous affichent les paramètres du système SAP permettant d'accéder aux informations sur l'entreprise que vous souhaitez lire dans **Talend Open Studio** à l'aide du composant **tSAPInput**.

- Créez une connexion au système SAP à l'aide de l'assistant SAP. Dans ce scénario, la connexion SAP est appelée **sap**.
- Appelez la fonction BAPI appelée *BAPI_COMPANY_GETDETAIL* à l'aide de l'assistant SAP pour accéder au document html BAPI stocké dans le système SAP et consultez les informations sur l'entreprise.
- Dans le champ **Name filter**, saisissez *BAPI** et cliquez sur le bouton **Search** pour afficher toutes les fonctions BAPI disponibles.
- Sélectionnez *BAPI_COMPANY_GETDETAIL* pour afficher le schéma décrivant les informations de l'entreprise.

La vue à droite de l'assistant affiche les métadonnées de la fonction *BAPI_COMPANY_GETDETAIL* et vous permet de configurer les paramètres nécessaires.

La vue **Document** affiche le document HTML SAP de la fonction *BAPI_COMPANY_GETDETAIL*.

La vue **Parameter** fournit les informations sur les paramètres d'entrée et de sortie nécessaire à la fonction *BAPI_COMPANY_GETDETAIL* pour retourner les valeurs.

- Dans la vue **Parameter**, cliquez sur l'onglet **Input** pour lister le ou les paramètre(s) d'entrée. Dans ce scénario, il n'y a qu'un seul paramètre d'entrée requis par la fonction *BAPI_COMPANY_GETDETAIL*. Ce paramètre s'appelle *COMPANYID*.

- Dans la vue **Parameter**, cliquez sur l'onglet **Output** pour lister les paramètres de sortie retournés par la fonction *BAPI_COMPANY_GETDETAIL*. Dans ce scénario, il y a deux paramètres de sortie : *COMPANY_DETAIL* et *RETURN*.

Chacun des paramètres de "structure" est composé de nombreux paramètres "single".

La vue **Test it** permet d'ajouter ou de supprimer des paramètres d'entrée en fonction de la fonction appelée. Dans ce scénario, l'objectif est de récupérer les métadonnées du paramètre "structure" appelé *COMPANY_DETAIL* qui est composé de 14 paramètres "single".

- Dans la colonne **Value** de la ligne **COMPANYID** du premier tableau, saisissez "*000001*" afin de retourner les données correspondant à l'entreprise dont l'identifiant est 000001.
- Dans la liste **Output type** en bas de l'assistant, sélectionnez **output.table**.
- Cliquez sur **Launch** en bas de la vue pour afficher la valeur de chaque paramètre "single" retourné par la fonction **BAPI_COMPANY_GETDETAIL**.
- Cliquez sur **Finish** pour fermer l'assistant et créer la connexion.

La connexion **sap** et la fonction **BAPI_COMPANY_GETDETAIL** apparaissent sous le noeud **SAP Connections** du **Repository**.

Pour récupérer les différents schémas de la fonction **BAPI_COMPANY_GETDETAIL** :

- Cliquez-droit sur **BAPI_COMPANY_GETDETAIL** dans le **Repository**.
- Sélectionnez **Retrieve schema** dans le menu.
- Dans l'assistant qui s'ouvre, sélectionnez les schémas que vous souhaitez récupérer. Ici, sélectionnez les deux schémas : **COMPANY_DETAIL** et **RETURN**.
- Cliquez sur **Next** pour afficher ces différents schémas.
- Cliquez sur **Finish**.

Les deux schémas apparaissent sous la fonction **BAPI_COMPANY_GETDETAIL** dans le **Repository.**

Dans ce scénario, l'objectif est de récupérer les données correspondant à l'entreprise dont l'identifiant est 000001 pour les afficher dans **Talend Open Studio** :

- Dans le **Repository**, sélectionnez la connexion SAP que vous venez de créer et glissez-la dans l'espace de modélisation, sélectionnez **tSAPConnection** dans la liste de composants et cliquez sur **OK**. Le composant **tSAPConnection** portant le nom de votre connexion, *sap*, sera ajouté à votre Job et ouvrira une connexion SAP.
- Double-cliquez sur le **tSAPConnection** pour afficher la vue **Component** :

- Si vous avez utiliser les métadonnées pour créer le composant, le mode **Repository** est sélectionné dans la liste **Property Type** et les champs suivants sont automatiquement renseignés. Si vous n'utilisez pas les métadonnées, sélectionnez le mode **Built-In** et renseignez les champs suivants manuellement.
- Dans le **Repository**, sélectionnez le schéma **BAPI_COMPANY_GETDETAIL** dans le dossier **sap**, et glissez-le dans l'espace de modélisation, sélectionnez **tSAPInput** dans la liste de composants et cliquez sur **OK**.
- Cliquez et déposez les composants **tFilterColumns** et **tLogRow** dans l'espace de modélisation.
- Connectez le **tSAPConnection** au **tSAPInput** via un lien **Trigger** > **OnSubJobOk**.
- Pour connecter le **tSAPInput** au **tLogRow**, cliquez-droit sur le **tSAPInput**, sélectionnez **Row** puis **row_COMPANY_DETAIL_1** dans le menu et cliquez sur le composant **tLogRow**.

- Dans l'espace de modélisation, sélectionnez le composant **tSAPInput** et cliquez sur la vue **Component** pour afficher ses propriétés.

Les propriétés du composant **tSAPInput** apparaissent automatiquement puisque le schéma est stocké dans le dossier **Metadata** et que le composant est initialisé par l'assistant SAP.

- Dans la vue **Component**, cochez la case **Use an existing connection** et sélectionnez le composant **tSAPConnection** dans la liste **Component list** pour utiliser la connexion ouverte par le composant de connexion.

Dans la zone **Initialize input**, vous pouvez voir les paramètres d'entrée nécessaires à la fonction *BAPI_COMPANY_GETDETAIL*.

Dans la zone **Outputs**, vous pouvez voir les différents schémas de la fonction **BAPI_COMPANY_GETDETAIL**, notamment le schéma **COMPANY_DETAIL** que vous récupérez en sortie.

- Dans l'espace de modélisation, sélectionnez le composant **tLogRow** et cliquez sur la vue **Component** pour paramétrer ses propriétés. Pour plus d'informations, consultez *tLogRow, page 1163*
- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

```
Starting job BAPI_SAP at 18:32 07/10/2009.

[statistics] connecting to socket on port 4100
[statistics] connected
.-----------------------------------.
|           #1. tLogRow_2           |
+---------------+-------------------+
| key           | value             |
+---------------+-------------------+
| COMPANY       | 000001            |
| NAME1         | Gesellschaft G00000 |
| NAME2         |                   |
| COUNTRY       | DE                |
| LANGU         | D                 |
| STREET        | Neurottstrasse 16 |
| PO_BOX        |                   |
| POSTL_COD1    | 69190             |
| CITY          | Walldorf          |
| CURRENCY      | EUR               |
| COUNTRY_ISO   | DE                |
| CURRENCY_ISO  | EUR               |
| LANGU_ISO     | DE                |
+---------------+-------------------+
[statistics] disconnected
Job BAPI_SAP ended at 18:32 07/10/2009. [exit code=0]
```

Le composant **tSAPInput** récupère du système SAP les métadonnées du paramètre *COMPANY_DETAIL* de type “structure” et le composant **tLogRow** affiche les informations dans la console.

Scénario 2 : Lire les données des différents schémas de la fonction RFC_READ_TABLE

Les composants SAP (**tSAPInput** et **tSAPOutput**) ainsi que l’assistant SAP sont basés sur des bibliothèques validées et fournies par SAP (JCO) permettant à l’utilisateur d’appeler les fonctions RFC ou BAPI et de récupérer les données du système SAP au niveau des tables, RFC ou BAPI.

Ce scénario utilise l’assistant SAP guidant l’utilisateur à travers les différentes étapes de création de connexions SAP et des fonctions RFC et BAPI. L’assistant SAP est uniquement disponible pour les utilisateurs **Talend Integration Suite**. Si vous êtes utilisateur de **Talend Open Studio** ou **Talend On Demand**, vous devez configurer les propriétés du composant **tSAPInput** manuellement.

Ce scénario Java utilise l’assistant SAP pour créer une connexion au système SAP et pour appeler une fonction RFC permettant de lire directement dans une table appelée *SFLIGHT* à partir du système SAP. Puis, il affiche dans **Talend Open Studio** les différentes structures de la table *SFLIGHT* stockée dans le système SAP.

- Créez une connexion au système SAP à l’aide de l’assistant SAP. Dans ce scénario, la connexion SAP est appelée **sap**.
- Appelez la fonction RFC appelée *RFC_READ_TABLE* à l’aide de l’assistant SAP pour accéder à la table du système SAP et pour visualiser sa structure.
- Dans le champ **Name filter**, saisissez *RFC** et cliquez sur le bouton **Search** pour afficher toutes les fonctions RFC disponibles.

- Sélectionnez *RFC_READ_TABLE* pour afficher les schémas de cette fonction.

La vue à droite de l'assistant affiche les métadonnées de la fonction *RFC_READ_TABLE* et permet de définir les paramètres nécessaires.

L'onglet **Document** affiche le document html SAP de la fonction *RFC_READ_TABLE*.

L'onglet **Parameter** fournit les informations sur les paramètres requis par la fonction *RFC_READ_TABLE* pour pouvoir retourner les valeurs des paramètres.

- Dans la vue **Parameter**, cliquez sur l'onglet **Table** pour obtenir une description de la structure des différentes tables de la fonction *RFC_READ_TABLE*.

Function: RFC_READ_TABLE
Purpose: External access to R/3 tables via RFC

Document | Parameter | Test it

Parameter type	Name	JCO type	Length	Value	Purpose:
table	DATA		0		
single	WA	CHAR	512		Character field length 512
table	FIELDS		0		
single	FIELDNAME	CHAR	30		Field name
single	OFFSET	NUM	6		Offset of a field in work area
single	LENGTH	NUM	6		Length (no. of characters)
single	TYPE	CHAR	1		ABAP data type (C,D,N,...)
single	FIELDTEXT	CHAR	60		Short text describing R/3 Rep
table	OPTIONS		0		
single	TEXT	CHAR	72		Text line of a message

Input | Output | Table

La vue **Test it** vous permet d'ajouter ou de supprimer des paramètres d'entrée en fonction de la fonction appelée. Dans ce scénario, l'objectif est de récupérer les différentes structures de la table *SFLIGHT*.

Function: RFC_READ_TABLE
Purpose: External access to R/3 tables via RFC

Document | Parameter | Test it

Name	Parameter Type	JCO type	Structure|Table	Length	Value	Purpose:
DELIMITER	input.single	CHAR		1	;	Sign for indicating field limits in DATA
NO_DATA	input.single	CHAR		1		If <> SPACE, only FIELDS is filled
QUERY_TABLE	input.single	CHAR		30	SFLIGHT	Table read
ROWCOUNT	input.single	INT		4		If <> SPACE, only FIELDS is filled
ROWSKIPS	input.single	INT		4		If <> SPACE, only FIELDS is filled

Add | Remove

Name	Parameter Type	JCO type	Structure|Table	Length	Value	Purpose:
WA	table.output	CHAR	DATA	512	000;LH ;0400;199...	Character field length 512
FIELDNAME	table.output	CHAR	FIELDS	30	MANDT	Field name
OFFSET	table.output	NUM	FIELDS	6	000000	Offset of a field in work...
LENGTH	table.output	NUM	FIELDS	6	000003	Length (no. of charact...
TYPE	table.output	CHAR	FIELDS	1	C	ABAP data type (C,D,N...
FIELDTEXT	table.output	CHAR	FIELDS	60	Client for WB train...	Short text describing R...
TEXT	table.output	CHAR	OPTIONS	72		Text line of a message

Add | Remove

Output type: output.table Constructure|Table: DATA

Launch

- Dans la colonne **Value** de la ligne **DELIMITER** du premier tableau, saisissez ";" comme séparateur de champ.
- Dans la colonne **Value** de la ligne **QUERY_TABLE**, saisissez *SFLIGHT* comme table à requêter.
- Dans la liste **Output type** en bas de l'assistant, sélectionnez **output.table**.
- Dans la liste **Constructure|Table**, sélectionnez **DATA**.

- Cliquez sur **Launch** en bas de la vue pour afficher les valeurs de paramètres retournées par la fonction *RFC_READ_TABLE*. Dans ce scénario, le délimiteur est “;” et la table à lire est *SFLIGHT*.
- Cliquez sur **Finish** pour fermer l’assistant et créer la connexion.

La connexion **sap** et la fonction **RFC_READ_TABLE** apparaissent sous le noeud **SAP Connections** du **Repository**.

Pour récupérer les différents schémas de la fonction **RFC_READ_TABLE** :

- Cliquez-droit sur **RFC_READ_TABLE** dans le **Repository**, et sélectionnez **Retrieve schema** dans le menu.
- Dans l’assistant qui s’ouvre, sélectionnez les schémas que vous souhaitez récupérer. Ici, sélectionnez les trois schémas : **DATA**, **FIELDS** et **OPTIONS**.
- Cliquez sur **Next** pour afficher ces différents schémas.
- Cliquez sur **Finish**.

Les trois schémas apparaissent sous la fonction **RFC_READ_TABLE** dans le **Repository.**

Dans ce scénario, l’objectif est de récupérer les données et le nom des colonnes de la table **SFLIGHT** pour les récupérer et les afficher dans **Talend Open Studio**. Ainsi, intéressez-vous uniquement au schéma **DATA** et **FIELDS** :

- Dans le **Repository**, sélectionnez la fonction **RFC_READ_TABLE** de la connexion **sap** et glissez-la dans l’espace de modélisation, sélectionnez **tSAPInput** dans la liste de composants et cliquez sur **OK**.
- Cliquez et déposez deux composants **tLogRow** dans l’espace de modélisation.

Pour connecter les composants entre eux :

- Cliquez-droit sur le **tSAPInput**, sélectionnez **Row** puis **row_DATA_1** dans le menu et cliquez sur le premier composant **tLogRow**.
- Cliquez-droit sur le **tSAPInput**, sélectionnez **Row** puis **row_FIELDS_1** dans le menu et cliquez sur le deuxième composant **tLogRow**.

- Dans l'espace de modélisation, double-cliquez sur **tSAPInput** pour afficher ses propriétés dans la vue **Component**.

Les propriétés du composant **tSAPInput** apparaissent automatiquement puisque le schéma est stocké dans le dossier **Metadata** et que le composant est initialisé par l'assistant SAP.

Dans la zone **Initialize input**, vous pouvez voir les paramètres d'entrée nécessaires pour la fonction *RFC_READ_TABLE*, à savoir le séparateur de champ ";" et le nom de la table **SFLIGHT**.

Dans la zone **Outputs**, vous pouvez voir les différents schémas de la table **SFLIGHT**, notamment **FIELDS** et **DATA** qui sont chacun dirigés vers des flux de sortie différents.

- Dans l'espace de modélisation, sélectionnez les composants **tLogRow** et cliquez sur la vue **Component** pour paramétrer leurs propriétés. Pour plus d'informations, consultez *tLogRow, page 1163*
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job jobsap at 16:13 01/10/2009.

[statistics] connecting to socket on port 4093
[statistics] connected
.----------+------+------+----+-----------------------------------------.
|                              tLogRow_3                                 |
|=---------+------+------+----+-----------------------------------------=|
|FIELDNAME |OFFSET|LENGTH|TYPE|FIELDTEXT                                |
|=---------+------+------+----+-----------------------------------------=|
|MANDT     |0     |3     |C   |Client for WB train. data model BC_Travel|
|CARRID    |4     |3     |C   |Airline carrier ID                       |
|CONNID    |8     |4     |N   |Flight connection Id                     |
|FLDATE    |13    |8     |D   |Flight date                              |
|PRICE     |22    |15    |P   |Airfare                                  |
|CURRENCY  |38    |5     |C   |Local currency of airline                |
|PLANETYPE |44    |10    |C   |Plane type                               |
|SEATSMAX  |55    |10    |X   |Maximum capacity                         |
|SEATSOCC  |66    |10    |X   |Occupied seats                           |
|PAYMENTSUM|77    |17    |P   |Total of current bookings                |
'----------+------+------+----+-----------------------------------------'

.-------------------------------------------------------------------------.
|                                tLogRow_2                                |
|=------------------------------------------------------------------------=|
|WA                                                                       |
|=------------------------------------------------------------------------=|
|000;LH ;0400;19950228; 899.00 ;DEM  ;A319      ;5E010000  ;03000000  ; 2639.00|
|000;LH ;0454;19951117;1499.00 ;DEM  ;A319      ;5E010000  ;02000000  ; 2949.00|
|000;LH ;0455;19950606;1090.00 ;USD  ;A319      ;DC000000  ;01000000  ; 1499.00|
|000;LH ;3577;19950428;6000.00 ;LIT  ;A319      ;DC000000  ;01000000  ;  600.00|
|000;SQ ;0026;19950228; 849.00 ;DEM  ;DC-10-10  ;7C010000  ;02000000  ; 1684.00|
'-------------------------------------------------------------------------'

[statistics] disconnected
Job jobsap ended at 16:13 01/10/2009. [exit code=0]
```

Le composant **tSAPInput** récupère le nom des colonnes de la table *SFLIGHT* et les données correspondantes à partir du système SAP et les composants **tLogRow** les affichent sous forme de table dans la console.

tSAPOutput

Propriétés du tSAPOutput

Famille de composant	Business	
Fonction	Le composant tSAPOutput écrit dans un système SAP.	
Objectif	Il permet d'écrire des données dans un système SAP.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier où sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant connexion de base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Use an existing connection	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient le Job parent et le Job enfant, la liste **Component list** présente uniquement les composants de connexion du même niveau de Job. Si vous souhaitez utiliser une connexion existant dans un autre niveau de Job, vous pouvez utiliser l'onglet **Dynamic settings**, afin de partager la connexion en question. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct. Pour plus d'information concernant l'onglet **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Connection configuration	**Client type** : Saisissez votre code de connexion SAP habituel. **Userid :** Saisissez l'identifiant de connexion utilisateur. **Password** : Saisissez le mot de passe. **Language** : Spécifiez la langue. **Host name** : Saisissez l'adresse IP du système SAP. **System number :** Saisissez le numéro du système.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous effectuez des modifications, le schéma passera automatiquement en built-in.
	FunName	Saisissez le nom de la fonction que vous souhaitez utiliser pour récupérer les données.
	Mapping	Configurez les paramètres afin de sélectionner les données à écrire dans le système SAP.
Advanced settings	*Release Connection*	Décochez cette case afin de continuer à utiliser la connexion sélectionnée, une fois que le composant a exécuté sa tâche.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé en tant que composant de sortie et requiert un composant d'entrée.	
Limitation	n/a	

Scénarios associés

Pour des scénarios associés, consultez les *Scénario 1 : Récupérer les métadonnées d'un système SAP, page cxi* et *Scénario 2 : Lire les données des différents schémas de la fonction RFC_READ_TABLE, page cxvii*.

tSAPRollback

Propriétés du tSAPRollback

Ce composant est étroitement lié aux composants **tSAPCommit** et **tSAPConnection**. Ils sont généralement utilisés ensemble lors de transactions..

Component family	Business/SAP	
Function	**tSAPRollback** annule la transaction dans le SAP connecté.	
Purpose	**tSAPRollback** évite le commit de transaction involontaire.	
Basic settings	*SAPConnection Component list*	Sélectionnez le composant de connexion **tSAPConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Release Connection	Décochez cette case afin de continuer à utiliser la connexion sélectionnée, une fois que le composan a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Usage	Ce composant est généralement utilisé avec des composants SAP et notamment avec les composants **tSAPConnection** et **tSAPCommit**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au **tSAPRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522* du composant **tMysqlRollback**.

tSugarCRMInput

Propriétés du tSugarCRMInput

Famille du composant	Business	
Fonction	Le **tSugarCRMInput** se connecte à un module de la base de données SugarCRM via le service Web adéquat.	
Objectif	Le **tSugarCRMInput** permet d'extraire les données d'une base de données SugarCRM à l'aide d'une requête.	
Basic settings	*SugarCRM Webservice URL*	Saisissez l'URL du service Web permettant de se connecter à la base de données SugarCRM.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Module	Sélectionnez le module adéquat dans la liste. Pour utiliser des tables personnalisées, sélectionnez **Use custom module** dans la liste . Les champs **Custom module package name** et **Custom module name** qui s'affichent sont automatiquement remplis avec les noms correspondants.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champ qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.. Pour ce composant, le schéma dépend du **Module** sélectionné.
	Query condition	Saisissez la requête permettant de sélectionner les données à extraire. Exemple : account_name= 'Talend'
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé en tant que composant de début. Un composant de sortie est nécessaire.	
Limitation	n/a	

Scénario: Extraire les données d'un compte à partir de SugarCRM

Ce scénario décrit un Job à deux composants visant à extraire les informations d'un compte à partir d'une base de données SugarCRM vers un fichier de sortie Excel.

- A partir de la Palette, cliquez-déposez un composant **tSugarCRMInput** et un composant **tFileOutputExcel**.
- Connectez les composants d'entrée et de sortie à l'aide d'un lien de type **Row Main**.
- Dans la vue **Component** du **tSugarCRMInput**, remplissez les informations de connexion dans les champs **SugarCRM Web Service URL**, **Username** et **Password**.
- Sélectionnez ensuite le module adéquat parmi ceux proposés dans la liste **Module**. Dans cet exemple, c'est *Accounts* qui est sélectionné.

- Le champ **Schema** est automatiquement renseigné en fonction du module sélectionné, mais il est possible de le modifier et de supprimer les colonnes dont vous n'avez pas besoin en sortie.
- Dans le champ **Query Condition**, saisissez la requête que vous voulez utiliser pour extraire les données du CRM. Dans cet exemple : "billing_address_city='Sunnyvale'"
- Sélectionnez ensuite le composant **tFileOutputExcel**.

- Renseignez le nom du fichier de destination dans le champ **File Name** ainsi que le nom de la feuille de calcul dans le champ **Sheet** puis cochez la case **Include header** pour considérer les entêtes.
- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

	A	B	C	D	E	F
1	id	name	account_t	industry	billing_ad	billing_address_
2	2ee60a11	T-Cat Media Group Inc 877900	Customer	Environmental	777 West	Sunnyvale
3	481bf585-	TJ O'Rourke Inc 323225	Customer	Insurance	999 Baker	Sunnyvale
4	7bd02e6c	CONS TRUST (AZ) 222552	Customer	Energy	67321 We	Sunnyvale
5	b2a4c25f-	CONS TRUST (AZ) 240011	Customer	Communications	345 Sugar	Sunnyvale
6	df5e078d-	2 Tall Stores 792551	Customer	Transportation	321 Unive	Sunnyvale
7	ef67c0f4-9	JAB Funds Ltd. 106774	Customer	Engineering	345 Sugar	Sunnyvale
8						
9						
10						

accounts

Sheet 1 / 1 | PageStyle_accounts | 100% | STD | *

Les données filtrées sont inscrites en sortie dans la feuille de calcul du fichier Excel spécifié.

tSugarCRMOutput

Propriétés du tSugarCRMOutput

Famille du composant	Business	
Fonction	Le composant **tSugarCRMOutput** écrit dans un module de la base de données SugarCRM via le service Web adéquat.	
Objectif	Le **tSugarCRMOutput** permet d'écrire des données dans une base de données SugarCRM.	
Basic settings	*SugarCRM WebService URL*	Saisissez l'URL du service Web permettant de se connecter à la base de données SugarCRM.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Module	Sélectionnez le module adéquat dans la liste. Pour utiliser des tables personnalisées, sélectionnez **Use custom module** dans la liste . Les champs **Custom module package name** et **Custom module name** qui s'affichent sont automatiquement remplis avec les noms correspondants.
	Action	Vous pouvez effectuer l'une des opérations suivantes sur les données du module SugarCRM : **Insert** : insérer les données. **Update** : mettre les données à jour.
	Schema et *Edit schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champ qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.. Pour ce composant, le schéma dépend du **Module** sélectionné.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé comme composant de sortie. Il nécessite un composant d'entrée.	
Limitation	n/a	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tVtigerCRMInput

Propriétés du tVtigerCRMInput

Famille du composant	Business/tVtigerCRM	
Fonction	Le **tVtigerCRMInput** se connecte à un module de la base de données vTigerCRM via le service Web adéquat.	
Objectif	Le **tVtigerCRMInput** permet d'extraire les données d'une base de données tVtigerCRM.	
Basic settings		
Vtiger Version	Sélectionnez la version du Vtiger Web Services que vous souhaitez utiliser.	
Vtiger 5.0	*Server Address*	Saisissez l'adresse IP du serveur tVtigerCRM.
	Port	Saisissez le numéro de port d'accès au serveur.
	Vtiger Path	Saisissez le chemin d'accès au serveur
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Version	Renseignez la version de tVtigerCRM que vous utilisez.
	Module	Sélectionnez le module adéquat dans la liste.
	Method	Sélectionnez la méthode adéquate dans la liste. Cette méthode permet de déterminer l'action à appliquer au module tVtigerCRM sélectionné.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (**Built-in**) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.. Pour ce composant, le schéma dépend du **Module** sélectionné.
Vtiger 5.1	*Endpoint*	Saisissez l'URL du serveur Web invoqué.
	Username	Saisissez le nom d'utilisateur afin de vous connecter au vTigerCRM..
	Access key	Saisissez la clé d'accès pour le nom d'utilisateur.
	Action	Vous pouvez effectuer l'une des opérations suivantes sur les données du module SugarCRM : **Insert** : insérer les données. **Update** : mettre les données à jour.
	Module	Sélectionnez le module adéquat dans la liste.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (**Built-in**) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.. Pour ce composant, le schéma dépend du **Module** sélectionné.
	Query condition	Saisissez la requête permettant de sélectionner les données à extraire.
	Manual input of SQL query	Cochez cette case pour saisir manuellement la requête que vous souhaitez effectuer dans le champ **Query**.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé en tant que composant de début. Un composant de sortie est nécessaire.	
Limitation	n/a	

Scénario associé

Aucun scénario n’est disponible pour ce composant.

tVtigerCRMOutput

Propriétés du tVtigerCRMOutput

Famille du composant	Business/vTigerCRM	
Fonction	Le composant **tVtigerCRMOutput** écrit dans un module de la base de données vTigerCRM via le service Web adéquat.	
Objectif	Le **tVtigerCRMOutput** permet d'écrire des données dans une base de données tVtigerCRM.	
Basic settings		
Vtiger Version	Sélectionnez la version du Vtiger Web Services que vous souhaitez utiliser.	
Vtiger 5.0	*Server Address*	Saisissez l'adresse IP du serveur tVtigerCRM.
	Port	Saisissez le numéro de port d'accès au serveur.
	Vtiger Path	Saisissez le chemin d'accès au serveur.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur au service Web.
	Version	Renseignez la version de tVtigerCRM que vous utilisez.
	Module	Sélectionnez le module adéquat dans la liste.
	Method	Sélectionnez la méthode adéquate dans la liste. Cette méthode permet de déterminer l'action à appliquer au module tVtigerCRM sélectionné.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (**Built-in**) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.. Pour ce composant, le schéma dépend du **Module** sélectionné.
Vtiger 5.1	*Endpoint*	Saisissez l'URL du serveur Web invoqué.
	Username	Saisissez le nom d'utilisateur afin de vous connecter au vTigerCRM..
	Access key	Saisissez la clé d'accès pour le nom d'utilisateur.
	Action	Vous pouvez effectuer l'une des opérations suivantes sur les données du module SugarCRM : **Insert** : insérer les données. **Update** : mettre les données à jour.
	Module	Sélectionnez le module adéquat dans la liste.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (**Built-in**) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.. Pour ce composant, le schéma dépend du **Module** sélectionné.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé comme composant de sortie. Il nécessite un composant d'entrée.	
Limitation	n/a	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

Composants Business Intelligence

Ce chapitre passe en revue les principaux composants présents dans le groupe **Business Intelligence** de la **Palette** de **Talend Open Studio**.

La famille **BI** regroupe tous les connecteurs couvrant des besoins tels que la lecture et l'écriture de bases de données multidimensionnelles ou OLAP, la génération de rapports Jasper, le suivi des modifications de bases de données à l'aide de tables Slow Changing Dimension, etc.

tDB2SCD

Propriétés du tDB2SCD

Famille du composant	Databases/Business Intelligence/DB2	
Fonction	Le composant **tDB2SCD** reflète et traque les modifications d'une table DB2 SCD dédiée.	
Objectif	Le **tDB2SCD** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table SCD dédiée.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Property type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Table schema	Nom du schéma de la base de données.

	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.
	SCD Editor	L'éditeur SCD Editor permet de constuire et de configurer les données du flux de sortie vers la table Slowly Changing Dimension. Pour plus d'informations, consultez la section *Méthodologie de gestion du SCD, page 20*.
	Use memory saving Mode	Cochez cette case pour améliorer les performances du système.
	Die on error	Cette case est décochée par défaut, ce qui vous permet de terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
	Debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
Utilisation	Ce composant est un composant de sortie. Par conséquent, il requiert un composant et une connexion de type Row Main en entrée.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23*.

tDB2SCDELT

Propriétés du tDB2SCDELT

Famille du composant	Databases/Business Intelligence/DB2	
Fonction	Le composant **tDB2SCDELT** reflète et traque les modifications d'une table DB2 SCD dédiée.	
Objectif	Le **tDB2SCDELT** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table DB2 SCD dédiée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tDB2Connection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Source table	Nom de la table contenant les données à filtrer.

	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrire à la fois pour que l'opération d'insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Surrogate Key	Sélectionnez dans la liste une colonne à utiliser comme clé de substitution.
	Creation	Sélectionnez la méthode à utiliser pour générer la clé de substitution.
	Source Keys	Sélectionnez une colonne ou plus à utiliser en tant que clé(s) pour assurer l'unicité des données entrantes.
	Use SCD Type 1 fields	Utilisez le type 1 si vous n'avez pas besoin de traquer les modifications, pour des corrections typographiques par exemple. Sélectionnez les colonnes du schéma qui servira de référence pour les modifications.

	Use SCD Type 2 fields	Utilisez le type 2 si vous avez besoin de traquer les modifications, pour garder une trace des mises à jour effectuées par exemple. Sélectionnez les colonnes du schéma qui servira de référence pour les modifications. **Start date** : Ajoute une colonne à votre schéma SCD pour déterminer la valeur de la date de départ. Vous pouvez sélectionner l'une des colonnes d'entrée du schéma comme date de départ (**Start Date**) dans la table SCD. **End Date** : Ajoute une colonne à votre schéma SCD pour déterminer la valeur de la date de fin pour le journal. Lorsque le journal est en mode actif, la colonne **End Date** a une valeur nulle ; pour éviter cela, vous pouvez sélectionner l'option **Fixed Year value** et saisir une année fictive. **Log Active Status** : Ajoute une colonne à votre schéma SCD pour renseigner les valeurs de statut **true** et **false**. Cette colonne permet de repérer facilement le journal actif. **Log versions** : Ajoute une colonne à votre schéma SCD pour renseigner le numéro de version du journal.
Advanced settings	*Debug mode*	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie et une connexion de type **Row Main**.	

Scénario associé

Pour un scénario associé, consultez les propriétés du *tDB2SCD, page 2*, ainsi que le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23*.

tGreenplumSCD

Propriétés du tGreenplumSCD

Famille du composant	Databases/Business Intelligence/Greenplum	
Fonction	Le composant **tGreenplumSCD** reflète et traque les modifications d'une table Greenplum SCD dédiée.	
Objectif	Le **tGreenplumSCD** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table SCD dédiée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez un tGreenplumConnection. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.

	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.
	SCD Editor	L'éditeur SCD Editor permet de constuire et de configurer les données du flux de sortie vers la table Slowly Changing Dimension. Pour plus d'informations, consultez la section *Méthodologie de gestion du SCD, page 20*.
	Use memory saving Mode	Cochez cette case pour améliorer les performances du système.
	Die on error	Cette case est décochée par défaut, ce qui vous permet de terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
	Debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
Utilisation	Ce composant est un composant de sortie. Par conséquent, il requiert un composant et une connexion de type Row Main en entrée.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23*.

tInformixSCD

Propriétés du tInformixSCD

Component family	Databases/Business Intelligence/Informix	
Fonction	Le composant **tInformixSCD** reflète et traque les modifications d'une table Informix SCD dédiée.	
Objectif	Le **tInformixSCD** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table SCD dédiée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma de la base de données.

	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.
	Instance	Nom de l'instance Informix à utiliser. Cette information se trouve généralement dans le fichier *SQL hosts*.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.
	SCD Editor	L'éditeur SCD Editor permet de constuire et de configurer les données du flux de sortie vers la table Slowly Changing Dimension. Pour plus d'informations, consultez la section *Méthodologie de gestion du SCD, page 20*.
	Use memory saving Mode	Cochez cette case pour améliorer les performances du système.
	Use Transaction	Décochez cette case lorsque la base de données est configurée en mode NO_LOG.
	Die on error	Cette case est décochée par défaut, ce qui vous permet de terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
	Debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
Usage	Ce composant est un composant de sortie. Par conséquent, il requiert un composant et une connexion de type Row Main en entrée.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23* du composant **tMysqlSCD**.

tIngresSCD

Propriétés du tIngresSCD

Component family	Databases/Business Intelligence/Ingres	
Fonction	Le composant **tIngresSCD** reflète et traque les modifications d'une table Ingres SCD dédiée.	
Objectif	Le **tIngresSCD** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table SCD dédiée.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Property type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Server	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.

	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.
	SCD Editor	L'éditeur SCD Editor permet de constuire et de configurer les données du flux de sortie vers la table Slowly Changing Dimension. Pour plus d'informations, consultez la section *Méthodologie de gestion du SCD, page 20*.
	Use memory saving Mode	Cochez cette case pour améliorer les performances du système.
	Die on error	Cette case est décochée par défaut, ce qui vous permet de terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
	Debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
Usage	Ce composant est un composant de sortie. Par conséquent, il requiert un composant et une connexion de type Row Main en entrée.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23* du composant **tMysqlSCD**.

tMondrianInput

Propriétés du tMondrianInput

Famille du composant	Business Intelligence/OLAP Cube	
Fonction	Le composant **tMondrianInput** lit des données à partir de bases relationnelles et produit des groupes de données multidimensionnels reposant sur une requête MDX.	
Objectif	Le **tMondrianInput** exécute une expression multi-dimensionnelle (MDX) correspondant à la structure du groupe de données et au schéma défini. Puis il passe les données multidimensionnelles obtenues au composant suivant à l'aide d'un lien de type **Row Main**.	
Basic settings	*Mondrian Version*	Sélectionnez la version de Mondrian que vous utilisez.
	DB type	Sélectionnez le type de relation approprié pour relier les bases de données.
	Property type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Datasource	Chemin d'accès et nom du fichier contenant les données.
	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Catalog	Chemin d'accès au catalogue (structure de l'entrepôt de données).
	MDX Query	Saisissez votre requête MDX en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma et dans la structure de l'entrepôt de données.

	Encoding	Sélectionnez le type d'encodage à partir de la liste ou choisissez l'option **Custom** pour le définir manuellement. Ce champ doit obligatoirement être renseigné.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités des requêtes MDX pour des groupes de données multidimensionnels.	

Scénario : Tables en jointure croisée

Le présent Job permet d'extraire des groupes de données multidimensionnels à partir de tables de bases de données relationnelles stockées dans une base MySQL. Les données sont récupérées à l'aide d'une expression multidimensionnelle (requête MDX). Il est clair que vous devez connaître la structure de vos données, ou du moins disposer d'une description de la structure (catalogue) comme référence du groupe de données à récupérer dans ces différentes dimensions.

- A partir de la **Palette**, cliquez-déposez un composant **tMondrianInput** (famille **Business Intelligence**) ainsi qu'un composant de sortie **tLogRow** (famille **Logs & Errors**).
- Reliez le **tMondrianInput** au composant de sortie à l'aide d'un lien de type **Row Main**.
- Sélectionnez le composant **tMondrianInput** puis cliquez sur la vue **Component**.

- Dans le champ **DB Type**, sélectionnez la base de données relationnelle à utiliser avec Mondrian.
- Sélectionnez l'entrée du **Repository** correspondante dans le champ **Property type**, dans le cas où vous stockez les détails de connexion à vos bases de données de manière centralisée. Dans cet exemple, les propriétés sont en mode **Built-in**.
- Remplissez les détails de connexion à votre base de données : **Host**, **Port**, **Database**, **User Name** et **Password**.
- Dans le champ **Schema**, sélectionnez le schéma correspondant à partir du **Repository** dans le cas où il est stocké de manière centralisée. Dans cet exemple, le schéma doit reste à définir (mode **Built-in**).

tMondrianInput_1

Column	Key	Type	Nullable	Date Patt...	Length	Pre...	D...	Co...
media	☐	String	☐				""	
drink	☐	String	☐				""	
unit_sales	☐	Double	☑					
store_cost	☐	Double	☑					
store_sales	☐	Double	☑					

- La base de données relationnelle que l'on veut interroger contient cinq colonnes : *media*, *drink*, *unit_sales*, *store_cost* et *store_sales*.
- La requête doit permettre de récupérer le contenu de *unit_sales*, *store_cost* et de *store_sales* pour les différentes valeur de *media* / *drink* en utilisant une requête MDX comme dans l'exemple ci-dessous :

CrossJoin Example 1

The current slicer is **1997**.

		Unit Sales	Store Cost	Store Sales
Radio	Alcoholic Beverages	75	70.40	168.62
	Beverages	97	75.70	186.03
	Dairy	54	36.75	89.03
TV	Alcoholic Beverages	76	70.99	182.38
	Beverages	188	167.00	419.14
	Dairy	68	45.19	119.55
Sunday Paper	Alcoholic Beverages	148	128.97	316.88
	Beverages	197	161.81	399.58
	Dairy	85	54.75	140.27
Street Handout	Alcoholic Beverages	158	121.14	294.55
	Beverages	270	201.28	520.55
	Dairy	84	50.26	128.32

- A partir de l'onglet **Basic settings** du composant **tMondrianInput**, renseignez le chemin d'accès au catalogue de l'entrepôt de données dans le champ **Catalog**. Ce catalogue décrit la structure de l'entrepôt.
- Entrez ensuite la requête MDX de la manière suivante :

```
"select
  {[Measures].[Unit Sales], [Measures].[Store Cost],
[Measures].[Store Sales]} on columns,
  CrossJoin(
    { [Promotion Media].[All Media].[Radio],
      [Promotion Media].[All Media].[TV],
      [Promotion Media].[All Media].[Sunday Paper],
      [Promotion Media].[All Media].[Street Handout] },
    [Product].[All Products].[Drink].children) on rows
from Sales
where ([Time].[1997])"
```

- Sélectionnez enfin le type d'encodage à partir de la liste du champ **Encoding Type**.
- Sélectionnez le composant **tLogRow** puis cochez l'option **Print header** pour afficher le nom des colonnes sur la console.
- Enregistrez le Job puis appuyez sur **F6** pour l'exécuter.

```
Starting job Mondrian at 15:31 08/01/2008.
media   drink   unit sales  store cost  store sales
Promotion Media].[All Media].[Radio]|[Product].[All Products].[Drink].[Alcoholic
everages]|75.0|70.3977|168.62
Promotion Media].[All Media].[Radio]|[Product].[All Products].[Drink].[Beverages]|97.0|75.7016|186.03
Promotion Media].[All Media].[Radio]|[Product].[All Products].[Drink].[Dairy]|54.0|36.7474|89.03
Promotion Media].[All Media].[TV]|[Product].[All Products].[Drink].[Alcoholic
everages]|76.0|70.9895|182.38
Promotion Media].[All Media].[TV]|[Product].[All Products].[Drink].[Beverages]|188.0|167.0025|419.14
Promotion Media].[All Media].[TV]|[Product].[All Products].[Drink].[Dairy]|68.0|45.1909|119.55
Promotion Media].[All Media].[Sunday Paper]|[Product].[All Products].[Drink].[Alcoholic
everages]|148.0|128.9736|316.88
Promotion Media].[All Media].[Sunday Paper]|[Product].[All Products].[Drink].
Beverages]|197.0|161.8067|399.58
Promotion Media].[All Media].[Sunday Paper]|[Product].[All Products].[Drink].
Dairy]|85.0|54.746|140.27
Promotion Media].[All Media].[Street Handout]|[Product].[All Products].[Drink].[Alcoholic
everages]|158.0|121.1394|294.55
Promotion Media].[All Media].[Street Handout]|[Product].[All Products].[Drink].
Beverages]|270.0|201.2816|520.55
Promotion Media].[All Media].[Street Handout]|[Product].[All Products].[Drink].
Dairy]|84.0|50.2604|128.32
Job Mondrian ended at 15:31 08/01/2008. [exit code=0]
```

La console affiche le résultat de *unit_sales*, *store_cost* et *store_sales* pour chaque type de *Drink* (*Beverages*, *Dairy*, *Alcoholic beverages*) croisé avec chaque type de *media* (*TV*, *Sunday Paper*, *Street handout*) comme illustré ci-dessus.

tMSSqlSCD

Propriétés du tMSSqlSCD

Famille de composant	Databases/Business Intelligence/MSSQL Server	
Fonction	Le composant **tMSSqlSCD** reflète et traque les modifications d'une table SCD MSSQL dédiée.	
Objectif	Le **tMSSqlSCD** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table SCD dédiée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Server	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Schema	Nom du schéma de la base de données.
	Database	Nom de la base de données.

	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.
	SCD Editor	L'éditeur SCD Editor permet de constuire et de configurer les données du flux de sortie vers la table Slowly Changing Dimension. Pour plus d'informations, consultez la section *Méthodologie de gestion du SCD, page 20*.
	Use memory saving Mode	Cochez cette case pour améliorer les performances du système.
	Die on error	Cette case est décochée par défaut, ce qui vous permet de terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
	Debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
Utilisation	Ce composant est un composant de sortie. Par conséquent, il requiert un composant et une connexion de type Row Main en entrée.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23*.

tMysqlSCD

Propriétés du tMysqlSCD

Famille de composant	Databases/Business Intelligence/MySQL	
Fonction	Le composant **tMysqlSCD** reflète et traque les modifications d'une table SCD MySQL dédiée.	
Objectif	Le **tMysqlSCD** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table SCD dédiée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	DB Version	Sélectionnez la version de Mysql que vous utilisez.
	Use an existing connection	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.

	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist**: :créer la table si nécessaire.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	SCD Editor	L'éditeur SCD Editor permet de constuire et de configurer les données du flux de sortie vers la table Slowly Changing Dimension. Pour plus d'informations, consultez la section *Méthodologie de gestion du SCD, page 20*.
	Use memory saving Mode	Cochez cette case pour améliorer les performances du système.
	Die on error	Cette case est décochée par défaut, ce qui vous permet de terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
	Debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
Utilisation	Ce composant est un composant de sortie. Par conséquent, il requiert un composant et une connexion de type Row Main en entrée.	

Méthodologie de gestion du SCD

Lors de dimensions à évolution lente ou Slowly Changing Dimensions (SCD), les données changent lentement. L'éditeur SCD permet de construire de manière simple le flux de sortie des données SCD. Dans l'éditeur SCD, vous pouvez relier les colonnes, sélectionner la clé de

substitution (Surrogate key) et paramétrer les attributs des colonnes modifiées en combinant les différents types de modifications SCD.

La figure ci-dessous représente l'éditeur SCD.

Clés SCD

Vous devez sélectionner une ou plusieurs colonnes qui serviront de clés permettant d'assurer l'unicité des données entrantes.
Vous devez aussi sélectionner une colonne sur laquelle positionner une clé de substitution (surrogate key) dans la table SCD et la relier à une des colonnes d'entrée de la table source. La valeur de la clé de substitution permet de relier les enregistrements de la table source aux enregistrements de la table SCD. L'éditeur utilise ce mapping pour localiser l'enregistrement dans la table SCD et pour déterminer si un enregistrement est nouveau ou s'il a été modifié. La clé de substitution est généralement la clé primaire de la table source, mais elle peut aussi correspondre à une autre clé à partir du moment où elle permet d'identifier de façon unique un enregistrement et où ses valeurs ne changent pas.

Source keys : Glissez une ou plusieurs colonnes du panneau **Unused** vers le panneau **Source keys**. Ces colonnes seront utilisées en tant que clé(s) assurant l'unicité des données entrantes.

Surrogate keys : Paramétrez la colonne dans laquelle la clé de substitution générée sera stockée. Une clé de substitution peut être générée en fonction de la méthode sélectionnée dans la liste **Creation**.

Creation : Sélectionnez une des méthodes suivantes permettant de générer la clé :

- **Auto increment** : la clé est auto-incrémentée.
- **Input field** : la clé est fournie par un champ d'entrée.
- **Routine** : vous pouvez accéder aux fonctions de base via le raccourci **Ctrl+ Espace**.
- **Table max +1** : la valeur maximum de la table SCD est incrémentée pour créer une clé de substitution.

Types de SCD

Il existe quatre types de dimensions à évolution lente (Slowly Changing Dimensions) : du **Type 0** au **Type 3**. Vous pouvez, d'un simple glisser-déposer, appliquer n'importe quel type de SCD à n'importe quelle colonne de la table source.

- **Type 0** : ce type de SCD n'est pas beaucoup utilisé. Certaines données dimensionnelles peuvent être écrasées et d'autres peuvent rester inchangées au cours du temps. Ce type de SCD convient lorsque aucun effort n'a été mis en place pour gérer les dimensions à caractère évolutif.
- **Type 1** : aucun historique n'est stocké dans la base de données. Les nouvelles données écrase les anciennes. Utilisez ce type si vous n'avez pas besoin de traquer les changements. Ce type de SCD convient lorsque vous corrigez certaines fautes, par exemple l'orthographe des mots.

- **Type2** : l'intégralité de l'historique est stockée dans la base de données. Ce type de SCD traque les données d'historique en enregistrant un nouvel enregistrement dans la table de dimension avec une nouvelle clé à chaque fois qu'un changement est effectué. Ce type de SCD convient lorsque l'on traque les mises à jour, par exemple.

  Le principe du SCD **Type 2** réside dans le fait qu'un nouvel enregistrement est ajouté à la table SCD lorsqu'un changement est détecté dans les colonnes sélectionnées. Notez que bien que plusieurs changements peuvent être effectués au même enregistrement sur plusieurs colonnes paramétrées en **Type 2**, une seule ligne traquant ces changements est ajoutée à la table SCD.

  Le schéma du SCD Type 2 devrait inclure des colonnes spécifiques au SCD contenant les informations de log standard, notamment :
  -start : ajoute une colonne au schéma de la table SCD contenant la date de début d'un enregistrement. Vous pouvez sélectionner une des colonnes du schéma d'entrée comme date de départ pour la table SCD.
  -end : ajoute une colonne au schéma de la table SCD contenant la date de fin d'un enregistrement. Lorsque l'enregistrement est en cours, la date de fin est de valeur **NULL** , sinon vous pouvez utiliser une année fixe en sélectionnant **Fixed Year Value** dans la liste et renseignez l'année fictive dans la cellule d'à coté pour éviter d'avoir une valeur nulle dans le champ **end**.
  -version : ajoute une colonne au schéma de la table SCD contenant le numéro de version de l'enregistrement.
  -active : ajoute une colonne au schéma de la table SCD contenant les statuts **true** ou **false**. Cette colonne permet de repérer facilement les enregistrements actifs.

- **Type 3** : seules les informations sur l'ancienne valeur d'une dimension est écrite dans la base de données. Ce type de SCD traque les changements en utilisant des colonnes séparées. Ce type de SCD convient lorsque l'on traque les valeurs précédentes d'une colonne qui change.

Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3

Ce scénario Java constitué de cinq composants décrit un Job traquant les modifications effectuées dans quatre des colonnes du fichier délimité source et écrit ces modifications, ainsi que l'historique des modifications dans une table SCD.

Le fichier délimité source contient différentes informations personnelles : des prénoms dans la colonne *firstname*, des noms dans la colonne *lastname*, des adresses dans la colonne *address*, des villes dans la colonne *city*, l'entreprise de la personne dans la colonne *company*, son âge dans la colonne *age* et son statut marital dans la colonne *status*. Une colonne *id* permet d'assurer l'unicité des données.

	A	B	C	D	E	F
1	id;firstname;lastname;address;city;company;age;status					
2	1;Janet;Anderson;511 Maple Ave. Apt. 1B;Agoura Hills;Adecco;30;married					
3	2;Harold;Smith;662 Lyons Circle;Alameda;Adidas;33;married					
4	3;Adam;Brown;220 Vine Ave.;Albany;Bridgestone;28;single					
5	4;Martin;Clinton;770 Exmoor Rd.;Alhambra;Cutco;25;single					
6	5;Sue;White;1486 Oakwood;Covina;Black&White;35;married					

Les changements de statut marital doivent écraser les anciens statuts enregistrés. Ce type de changement correspond à un changement SCD **Type 1**.

Un nouvel enregistrement dans la table dimensionnelle avec une clé différente sera inséré à chaque fois qu'une personne change d'entreprise. Ce type de changement correspond à un changement SCD **Type 2**.

Vous allez récupérer l'ancienne ville et l'ancienne adresse d'une personne. Ce type de changement correspond à un changement SCD **Type 3**.

Pour réaliser ce type de scénario, il est préférable de le séparer en trois étapes principales : la définition du flux principal du Job, le paramétrage de l'éditeur SCD et la création de la table de modification dans la base de données.

Etape 1 : Définir le flux principal du Job

- Cliquez et déposez les composants suivants dans l'éditeur graphique : **tMysqlConnection**, **tFileInputDelimited**, **tLogRow**, **tMysqlSCD**, **tMysqlCommit**.
- Connectez les composants **tFileInputDelimited**, **tLogRow** et **tMysqlSCD** à l'aide d'un lien **Row Main**. Ce sous-Job correspond au flux principal du Job.
- Connectez le **tMysqlConnection** au **tFileInputDelimited** et le **tMysqlSCD** au **tMysqlCommit** à l'aide de liens **OnComponentOk**.

- Dans l'éditeur graphique, double-cliquez sur le composant **tMysqlConnection** pour afficher ces propriétés de base dans l'onglet **Basic settings** et renseigner manuellement les informations de connexion. Lorsque plusieurs composants Database sont utilisés, le composant **tMysqlConnection** devrait être utilisé pour éviter de renseigner plusieurs fois les mêmes informations de connexion à la base de données.

  Si vous avez déjà stocké les informations de connexion dans le **Repository**, glissez l'élément **Metadata** souhaité dans l'éditeur graphique et les informations de connexion à la base de données apparaîtront automatiquement dans les champs correspondant. Pour plus d'informations sur les métadonnées, consultez *Centraliser les métadonnées* du **Guide Utilisateur** de **Talend Open Studio**.

Dans ce scénario, l'objectif est de se connecter à la table SCD dans laquelle toutes les modifications apportées au fichier délimité seront traquées.

- Dans l'éditeur graphique, double-cliquez sur le composant **tFileInputDelimited** pour afficher la vue **Basic settings**.

- Cliquez sur le bouton **[...]** à coté du champ **File Name** pour sélectionner le chemin d'accès au fichier délimité *dataset.csv* source contenant les informations personnelles.
- Dans les champs **Row** et **Field Separator**, renseignez respectivement les séparateurs de lignes et de champs utilisés dans le fichier source.

  Les champs **File Name**, **Row separator** et **Field separators** sont obligatoires.

- Si nécessaire, renseignez les champs **Header**, **Footer** et **Limit**.
  Dans ce scénario, saisissez *1* dans le champ **Header** puisque le fichier source contient un en-tête. Laissez le champ **Footer** à *0* puisque le fichier source ne contient pas de pied de page et ne renseignez pas le champ **Limit** déterminant le nombre de lignes devant être traitées.

- Cliquez sur **Edit schema** pour décrire la structure des données du fichier délimité source.

Dans ce scénario, le schéma source est composé de huit colonnes : *firstname*, *lastname*, *address*, *city*, *company*, *age*, *status* et *id*.

- Dans l'onglet **Basic settings** du composant **tLogRow**, sélectionnez le mode **Table** afin que le contenu du fichier source, ainsi que les attributs variants apparaissent dans les cellules du tableau dans la console avant d'être traitées par composant SCD.
- Dans l'éditeur graphique, double-cliquez sur le composant **tMysqlSCD** pour paramétrer ses propriétés.

- Dans l'onglet **Basic settings**, cochez la case **Use an existing connection** pour réutiliser les informations de connexion définies dans les propriétés du composant **tMysqlConnection**.
- Dans le champ **Table**, saisissez le nom de la table SCD dans laquelle les changements seront stockés.
- Cliquez sur **Sync columns** pour récupérer toutes les colonnes du fichier délimité.
- Si nécessaire, cliquez sur **Edit schema** pour vérifier la structure des données en sortie du composant **tMysqlSCD**.

- Dans l'éditeur graphique, double-cliquez sur le composant **tMysqlCommit** pour paramétrer ses propriétés.
- Dans la liste **Component list**, sélectionnez la connexion adéquate s'il y a plusieurs composants de connexion dans le Job.

Etape 2 : Paramétrer l'éditeur SCD

- Dans l'onglet **Basic settings** du composant **tMysqlSCD**, cliquez sur le bouton **[...]** à coté du champ **SCD Editor** pour ouvrir l'éditeur **SCD editor** et construire le flux de données en sortie du SCD.

Toutes les colonnes du fichier délimité source, provenant du composant précédent et n'ayant pas encore été utilisées, apparaissent dans le panneau **Unused** de l'éditeur **SCD**. Tous les autres panneaux de l'éditeur sont vides.

- A partir de la liste **Unused**, glissez la colonne *id* vers le panneau **Source keys** pour être utilisée comme clé permettant d'assurer l'unicité des données entrantes.
- Dans le panneau **Surrogate keys**, donnez un nom à la clé de substitution (surrogate key) dans le champ **Name**, **SK1** pour ce scénario.
- Dans la liste **Creation**, sélectionnez la méthode à utiliser pour générer la clé de substitution, **Auto-increment** pour ce scénario.
- Dans la liste **Unused**, glissez les colonnes *firstname* et *lastname* vers le panneau **Type 0**, les changements effectués dans ces colonnes ne sont pas pertinents dans cet exemple.
- Glissez la colonne *status* vers le panneau **Type 1**. L'ancienne valeur sera écrasée par la nouvelle.
- Glissez la colonne *company* vers le panneau **Type 2**. A chaque fois qu'une personne change d'entreprise, un nouvel enregistrement sera inséré dans la table dimensionnelle avec une nouvelle clé.
  Dans la zone **Versioning** :
  -Paramétrez les colonnes **start** et **end** de votre table SCD contenant respectivement les dates de début et de fin. La date de fin est nulle pour les enregistrements en cours jusqu'à ce qu'un changement soit détecté. A ce moment là, la date de fin est renseignée et un nouvel enregistrement est ajouté sans date de fin.
  Dans ce scénario, sélectionnez **Fixed Year Value** dans la colonne **end** et saisissez une année fictive pour éviter d'avoir une valeur nulle dans le champ de date de fin.
  -cochez le case **version** pour conserver le numéro de version de l'enregistrement.
  -cochez la case **active** pour ajouter la colonne **Status** contenant les statuts **True** ou **False** des enregistrements. **True** correspond à l'enregistrement en cours et **False** correspond à l'enregistrement modifié.
- Glissez les colonnes *address* et *city* vers le panneau **Type 3** pour ne récupérer que les informations sur les anciennes valeurs des adresses et des villes.

Pour plus d'informations sur les types de SCD, consultez *Méthodologie de gestion du SCD, page 20.*.

- Cliquez sur **Ok** pour valider votre configuration et fermer l'éditeur **SCD**.
- Cliquez sur **Edit schema** pour afficher le flux de données en entrée et en sortie. Le schéma de sortie du SCD devrait inclure les colonnes spécifiques au SCD contenant les informations de log standard définies dans l'éditeur **SCD editor**.

Etape 3 : Créer la table SCD

Dans la base de données, vous devez créer la table SCD récupérant les changements comme définis dans le composant **tMysqlSCD**. La table SCD doit être composée du schéma du fichier délimité et du schéma de sortie SCD.

Dans ce scénario, la table SCD est structurée selon le schéma suivant : *firstname, lastname, address, city, company, age, status, id, SK1, previous_address, previuos city, scd_active, scd_end, scd version* et *scd_date*.

Pour écrire les changements et l'historique de ces changements dans la table SCD sélectionnée, enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job SCD1 at 00:07 15/12/2008.
.--+---------+--------+----------------------+------------+-----------+---+-------.
|                                   tLogRow_1                                      |
|=-+---------+--------+----------------------+------------+-----------+---+------=|
|id|firstname|lastname|address               |city        |company    |age|status |
|=-+---------+--------+----------------------+------------+-----------+---+------=|
|1 |Janet    |Anderson|511 Maple Ave.  Apt. 1B|Agoura Hills|Adecco     |30 |married|
|2 |Harold   |Smith   |662 Lyons Circle      |Alameda     |Adidas     |33 |married|
|3 |Adam     |Brown   |220 Vine Ave.         |Albany      |Bridgestone|28 |single |
|4 |Martin   |Clinton |770 Exmoor Rd.        |Alhambra    |Cutco      |25 |single |
|5 |Sue      |White   |1486 Oakwood          |Covina      |Black&White|35 |married|
'--+---------+--------+----------------------+------------+-----------+---+-------'

Job SCD1 ended at 00:07 15/12/2008. [exit code=0]
```

La console affiche le contenu du fichier délimité d'entrée.

Janet divorce, déménage à *Adelanto* au *355 Golf Rd* et travaille maintenant à *Greenwood.*

Adam s'est marrié, a déménagé à *Belmont* au *2505 Alisson ct.* et travaille à *Scoop.*

Mettez à jour le fichier délimité avec ces informations et appuyez sur **F6** pour exécuter le Job.

```
Starting job SCD1 at 00:32 15/12/2008.
.--+---------+--------+----------------+--------+-----------+---+-------.
|                                tLogRow_1                                |
|=-+---------+--------+----------------+--------+-----------+---+------=|
|id|firstname|lastname|address         |city    |company    |age|status |
|=-+---------+--------+----------------+--------+-----------+---+------=|
|1 |Janet    |Anderson|355 Golf Rd     |Adelanto|Greenwood  |30 |single |
|2 |Harold   |Smith   |662 Lyons Circle|Alameda |Adidas     |33 |married|
|3 |Adam     |Brown   |2505 Alisson ct |Belmont |Scoop      |28 |married|
|4 |Martin   |Clinton |770 Exmoor Rd.  |Alhambra|Cutco      |25 |single |
|5 |Sue      |White   |1486 Oakwood    |Covina  |Black&White|35 |married|
'--+---------+--------+----------------+--------+-----------+---+-------'

Job SCD1 ended at 00:32 15/12/2008. [exit code=0]
```

La console affiche les changements apportés aux informations personnelles et la table SCD affiche l'historique de ces changements, ainsi que leur statut et leur numéro de version.

SK1	id	firstname	lastname	address	previous_address	city	previous_city	company	status	scd_version	scd_activ
1	1	Janet	Anderson	511 Maple Ave. Apt. 1B	511 Maple Ave. Apt. 1B	Agoura Hills	Adelanto	Adecco	single	1	
2	2	Harold	Smith	662 Lyons Circle	NULL	Alameda	NULL	Adidas	married	1	
3	3	Adam	Brown	220 Vine Ave.	220 Vine Ave.	Albany	Belmont	Bridgestone	married	1	
4	4	Martin	Clinton	770 Exmoor Rd.	NULL	Alhambra	NULL	Cutco	single	1	
5	5	Sue	White	1486 Oakwood	NULL	Covina	NULL	Black&Wh...	married	1	
6	1	Janet	Anderson	355 Golf Rd	511 Maple Ave. Apt. 1B	Adelanto	Agoura Hills	Greenwood	single	2	
7	3	Adam	Brown	2505 Alisson ct	220 Vine Ave.	Belmont	Albany	Scoop	married	2	

tMysqlSCDELT

Propriétés du tMysqlSCDELT

Famille du composant	Databases/Business Intelligence/MySQL	
Fonction	Le composant **tMysqlSCDELT** reflète et traque les modifications d'une table MySQL SCD dédiée.	
Objectif	Le **tMysqlSCDELT** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table MySQL SCD dédiée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	DB Version	Sélectionnez la version de Mysql que vous utilisez.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tMysqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Source table	Nom de la table contenant les données à filtrer.

	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrire à la fois pour que l'opération d'insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Source Keys	Sélectionnez une colonne ou plus à utiliser en tant que clé(s) pour assurer l'unicité des données entrantes.
	Use SCD Type 1 fields	Utilisez le type 1 si vous n'avez pas besoin de traquer les modifications, pour des corrections typographiques par exemple. Sélectionnez les colonnes du schéma qui servira de référence pour les modifications.

	Use SCD Type 2 fields	Utilisez le type 2 si vous avez besoin de traquer les modifications, pour garder une trace des mises à jour effectuées par exemple. Sélectionnez les colonnes du schéma qui servira de référence pour les modifications. **Start date** : Ajoute une colonne à votre schéma SCD pour déterminer la valeur de la date de départ. Vous pouvez sélectionner l'une des colonnes d'entrée du schéma comme date de départ (Start Date) dans la table SCD. **End Date** : Ajoute une colonne à votre schéma SCD pour déterminer la valeur de la date de fin pour le journal. Lorsque le journal est en mode actif, la colonne **End Date** a une valeur nulle ; pour éviter cela, vous pouvez sélectionner l'option **Fixed Year value** et saisir une année fictive. **Log Active Status** : Ajoute une colonne à votre schéma SCD pour renseigner les valeurs de statut **true** et **false**. Cette colonne permet de repérer facilement le journal actif. **Log versions** : Ajoute une colonne à votre schéma SCD pour renseigner le numéro de version du journal.
Advanced settings	*Debug mode*	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie et une connexion de type **Row Main**.	

Scénario associé

Pour un scénario associé, consultez les propriétés du *tMysqlSCD, page 19*, ainsi que le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23*.

tOracleSCD

Propriétés du tOracleSCD

Famille de composant	Databases/Business Intelligence/Oracle	
Fonction	Le composant **tOracleSCD** reflète et traque les modifications d'une table SCD Oracle dédiée.	
Objectif	Le **tOracleSCD** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table SCD dédiée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Connection type	Liste déroulante des pilotes disponibles.
	DB Version	Sélectionnez la version d'Oracle que vous utilisez.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.

	Database	Nom de la base de données.
	Schema	Nom du schéma de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.
	SCD Editor	L'éditeur SCD Editor permet de constuire et de configurer les données du flux de sortie vers la table Slowly Changing Dimension. Pour plus d'informations, consultez la section *Méthodologie de gestion du SCD, page 20*.
	Use memory saving Mode	Cochez cette case pour améliorer les performances du système.
	Die on error	Cette case est décochée par défaut, ce qui vous permet de terminer le traitement avec les lignes sans erreur, et d'ignorer les lignes en erreur.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
	Debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
Utilisation	Ce composant est un composant de sortie. Par conséquent, il requiert un composant et une connexion de type Row Main en entrée.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23*.

tOracleSCDELT

Propriétés du tOracleSCDELT

Famille du composant	Databases/Business Intelligence/Oracle	
Fonction	Le composant **tOracleSCDELT** reflète et traque les modifications d'une table Oracle SCD dédiée.	
Objectif	Le **tOracleSCDELT** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table Oracle SCD dédiée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tOracleConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Connection type	Liste déroulante des pilotes disponibles.
	DB Version	Sélectionnez la version d'Oracle que vous utilisez.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.

	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Source table	Nom de la table contenant les données à filtrer.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrire à la fois pour que l'opération d'insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Surrogate Key	Sélectionnez dans la liste une colonne à utiliser comme clé de substitution.
	Creation	Sélectionnez la méthode à utiliser pour générer la clé de substitution.
	Source Keys	Sélectionnez une colonne ou plus à utiliser en tant que clé(s) pour assurer l'unicité des données entrantes.
	Use SCD Type 1 fields	Utilisez le type 1 si vous n'avez pas besoin de traquer les modifications, pour des corrections typographiques par exemple. Sélectionnez les colonnes du schéma qui servira de référence pour les modifications.

	Use SCD Type 2 fields	Utilisez le type 2 si vous avez besoin de traquer les modifications, pour garder une trace des mises à jour effectuées par exemple. Sélectionnez les colonnes du schéma qui servira de référence pour les modifications. **Start date** : Ajoute une colonne à votre schéma SCD pour déterminer la valeur de la date de départ. Vous pouvez sélectionner l'une des colonnes d'entrée du schéma comme date de départ (Start Date) dans la table SCD. **End Date** : Ajoute une colonne à votre schéma SCD pour déterminer la valeur de la date de fin pour le journal. Lorsque le journal est en mode actif, la colonne **End Date** a une valeur nulle ; pour éviter cela, vous pouvez sélectionner l'option **Fixed Year value** et saisir une année fictive. **Log Active Status** : Ajoute une colonne à votre schéma SCD pour renseigner les valeurs de statut **true** et **false** . Cette colonne permet de repérer facilement le journal actif. **Log versions** : Ajoute une colonne à votre schéma SCD pour renseigner le numéro de version du journal.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie et une connexion de type **Row Main**.	

Scénario associé

Pour un scénario associé, consultez les propriétés du *tOracleSCD, page 35*, ainsi que le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23*.

tPaloCheckElements

Propriétés du tPaloCheckElements

Famille de composant	Business Intelligence/Cube OLAP/Palo	
Fonction	Le composant **tPaloCheckElements** vérifie l'existence d'éléments du flux de données d'entrée dans un cube donné.	
Objectif	Ce composant peut être utilisé avec le **tPaloOutputMulti**. Il vérifie si les éléments du flux d'entrée existent dans le cube donné, avant de les écrire. Il peut également définir une valeur par défaut qui peut être utilisée si les éléments n'existent pas.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
Connection configuration Indisponible lors de l'utilisation d'une connexion existante	*Host Name*	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Server Port	Saisissez le numéro du port d'écoute du serveur Palo.
	Username et *Password*	Informations d'authentification de l'utilisateur de Palo.
	Database	Saisissez le nom de la base de données dans laquelle écrire les données.

	Cube	Saisissez le nom du cube dans lequel les données doivent être écrites.
	On element error	Sélectionnez l'action à effectuer lorsqu'un élément n'existe pas. - **Reject row** : La ligne correspondante sera rejetée et placée dans le flux de rejet. - **Use default** : La valeur par défaut définie sera utilisée. - **Stop** : le processus complet sera interrompu.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.
		Un tableau vous permet de définir les éléments à vérifier. - **Column** : affiche la (les) colonne(s) du schéma d'entrée. Cette colonne est automatiquement remplie une fois qu'un schéma d'entrée est saisi ou créé. - **Element type** : sélectionnez le type d'élément de la colonne d'entrée. Seule une colonne peut être définie en tant que **Measure**. - **Default** : saisissez la valeur par défaut qui sera utilisée si l'option **Use default** a été sélectionnée dans le champ **On element error** .
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant requiert un composant d'entrée.	
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Rejects **Trigger** : Run if, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Rejects Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	Ce composant fonctionne uniquement avec des cubes Palo de type "Normal".	

Scénario associé

Pour un scénario associé, consultez le *Scénario 2 : Refuser un flux de données d'entrée lorsque les éléments à écrire n'existent pas dans un cube donné, page 85*.

tPaloConnection

Propriétés du tPaloConnection

Famille de composant	Business Intelligence/Cube OLAP/Palo	
Fonction	Le composant **tPaloConnection** ouvre une connexion vers un serveur Palo et la laisse ouverte durant le temps de traitement. Chaque autre composant Palo du processus peut utiliser cette connexion.	
Objectif	Ce composant permet le partage d'une connexion à un serveur Palo durant le processus complet.	
Basic settings	*Host Name*	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Server Port	Saisissez le numéro du port d'écoute du serveur Palo.
	Username et *Password*	Informations d'authentification de l'utilisateur de Palo.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé avec les composants Palo afin de partager une connexion à un serveur Palo.	
Connections		Liens de sortie (d'un composant à un autre) : **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur Talend Open Studio**.
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Créer une dimension avec des éléments, page 63*.

tPaloCube

Propriétés du tPaloCube

Famille de composant	Business Intelligence/Cube OLAP/Palo	
Fonction	Le composant **tPaloCube** crée, supprime ou vide des cubes Palo à partir de dimensions existantes dans une base de données Palo.	
Objectif	Ce composant effectue des opérations sur un cube Palo donné.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
Connection configuration Indisponible lors de l'utilisation d'une connexion existante.	*Host Name*	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Server Port	Saisissez le numéro du port d'écoute du serveur Palo.
	Username et *Password*	Informations d'authentification de l'utilisateur de Palo.
	Database	Saisissez le nom de la base de données dans laquelle l'opération donnée doit s'effectuer.
	Cube	Saisissez le nom du cube dans lequel l'opération donnée doit s'effectuer.

	Cube type	Sélectionnez le type de cube dans la liste déroulante pour le cube sur lequel l'opération donnée doit s'effectuer : - **Normal** : Type de cube "normal", type par défaut. - **Attribut** : un cube **Attribute** sera créé avec la création d'un cube "normal". - **User Info** : un cube **User Info** sera créé/modifié par ce composant.
	Action on cube	Sélectionnez l'opération que vous souhaitez effectuer sur le cube défini : - **Create cube** : le cube n'existe pas et sera créé. - **Create cube if not exists** : crée un cube s'il n'existe pas. - **Delete cube if exists and create** : supprime un cube s'il existe et en crée un nouveau. - **Delete cube** : supprime le cube de la base de données. - **Clear cube** : supprime les données du cube.
	Dimension list	Ajoutez des lignes en cliquant sur le bouton **[+]** et dans chaque ligne, saisissez le nom d'une dimension existant dans la base de données et devant être utilisée dans le cube. L'ordre de cette liste définit également l'ordre de création des dimensions.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en standalone dans la création dynamique de cubes, avec une liste définie de dimensions.	
Global Variables		**Cubename** : Indique le nom du cube traité. Disponible en tant que variable **After**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* du **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	Le cube créant un processus ne crée pas les dimensions à partir de rien. Les dimensions utilisées dans le cube ont été créées précédemment.	

Scénario : Créer un cube dans une base de données existante

Le Job de ce scénario crée un nouveau cube à deux dimensions dans la base de données démo *Biker* de Palo.

Pour reproduire ce scénario, procédez comme suit :

- Déposez un composant **tPaloCube** de la **Palette** dans l'espace de modélisation graphique.
- Double-cliquez sur le composant **tPaloCube** afin d'ouvrir sa vue **Component**.

- Dans le champ **Host name**, saisissez le nom de l'hôte ou l'adresse IP du serveur hôte, *localhost* dans cet exemple.
- Dans le champ **Server Port**, saisissez le numéro du port d'écoute du serveur Palo. Dans ce scénario, le numéro est *7777*.
- Dans les champs **Username** (identifiant) et **Password** (mot de passe), saisissez vos informations d'authentification. Dans cet exemple, les deux sont *admin*.

- Dans le champ **Database**, saisissez le nom de la base de données dans laquelle vous souhaitez créer le cube, *Biker* dans cet exemple.
- Dans le champ **Cube**, saisissez le nom que vous souhaitez utiliser afin de créer le cube, *bikerTalend*, par exemple.
- Dans le champ **Cube type**, sélectionnez le type **Normal** dans la liste déroulante pour le cube à créer, ce qui signifie que ce cube sera de type “Normal”, le type par défaut.
- Dans le champ **Action on cube**, sélectionnez l’action à effectuer. Dans ce scénario, sélectionnez **Create cube**.
- Sous le tableau **Dimensionlist**, cliquez deux fois sur le bouton **[+]** afin d’ajouter deux lignes.
- Dans le tableau **Dimensionlist**, saisissez le nom de chaque nouvelle ligne afin de remplacer le nom par défaut. Dans ce scénario, saisissez *Months* pour la première ligne et *Products* pour la seconde. Ces deux dimensions existent déjà dans la base de données *Biker* dans laquelle le nouveau cube sera créé.
- Appuyez sur **F6** afin d’exécuter le Job.

Un nouveau cube a été créé dans la base de données *Biker* et les deux dimensions sont ajoutées dans ce cube.

tPaloCubeList

Propriétés du tPaloCubeList

Famille de composant	Business Intelligence/Cube OLAP/Palo	
Fonction	Le composant **tPaloCubeList** récupère les informations d'une liste de cubes dans la base de données Palo déterminée.	
Objectif	Ce composant liste le nom des cubes, leur types, le nombre de dimensions assignées, le nombre de cellules remplies dans la base de données déterminée.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
Connection configuration Indisponible lors de l'utilisation d'une connexion existante.	*Host Name*	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Server Port	Saisissez le numéro du port d'écoute du serveur Palo.
	Username et *Password*	Informations d'authentification de l'utilisateur de Palo.
	Database	Saisissez le nom de la base de données contenant les cubes dont vous souhaitez récupérer les informations.

Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en tant que composant de début. Il requiert un composant de sortie.	
Global Variables		**Number of cubes** : indique le nombre de cubes traités dans la base de données déterminée. Disponible en tant que variable **After**. Retourne un entier. **Cube_ID** : indique les IDs des cubes traités dans la base de données déterminée. Disponible en tant que variable **Flow**. Retourne un entier. **Cubename** : indique le nom des cubes traités dans la base de données déterminée. Disponible en tant que variable **Flow**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* du **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	Le schéma de sortie est fixe et en lecture seule.	

Découvrir le schéma de sortie en lecture seule du tPaloCubeList

Le tableau ci-dessous présente les informations relatives au schéma en lecture seule du composant **tPaloCubeList**.

Colonne	Type	Description
Cube_id	int	ID interne du cube.
Cube_name	string	Nom du cube.
Cube_dimensions	int	Nombre de dimensions dans le cube.
Cube_cells	long	Nombrede cellules à l'intérieur du cube.
Cube_filled_cells	long	Nombre de cellules remplies dans le cube.
Cube_status	int	Statut du cube. Il peut être : - **0** : non chargé - **1** : chargé - **2** : changé

Colonne	Type	Description
Cube_type	int	Type du cube. Il peut être : - **0** : normal - **1** : system - **2** : attribute - **3** : user info - **4** . gpu type

Scénario : Récupérer des informations détaillées d'un cube d'une base de données déterminée

Le Job dans ce scénario récupère des informations détaillées concernant les cubes de la base de données démo de Palo *Biker*.

Pour reproduire ce scénario, procédez comme suit :

- Déposez un composant **tPaloCubeList** et un **tLogRow** de la **Palette** dans l'espace de modélisation graphique.
- Cliquez-droit sur le **tPaloCubeList** afin d'ouvrir son menu contextuel.
- Dans ce menu, sélectionnez **Row** > **Main** pour relier les deux composants.
- Double-cliquez sur le **tPaloCube** afin d'ouvrir sa vue **Component**.

- Dans le champ **Host name**, saisissez le nom de l'hôte ou l'adresse IP du serveur hôte, *localhost* dans cet exemple.
- Dans le champ **Server Port**, saisissez le numéro du port d'écoute du serveur Palo. Dans ce scénario, le numéro est *7777*.

- Dans les champs **Username** (identifiant) et **Password** (mot de passe), saisissez vos informations d'authentification. Dans cet exemple, les deux sont *admin*.
- Dans le champ **Database**, saisissez le nom de la base de données dans laquelle vous souhaitez créer le cube, *Biker* dans cet exemple.
- Appuyez sur **F6** pour exécuter le Job.

Les informations du cube sont récupérées de la base de données *Biker* et sont affichées dans la console de la vue **Run**.

```
[statistics] connected
26|Orders|8|45291236390400|133005|1|0
27|bikerTalend|2|11830|0|1|0
1|#_GROUP_CUBE_DATA|2|8|0|1|1
2|#_CONFIGURATION|1|2|2|1|1
2|#_CONFIGURATION|1|2|2|1|1
2|#_CONFIGURATION|1|2|2|1|1
6|#_SUBSET_LOCAL|3|0|0|1|1
6|#_SUBSET_LOCAL|3|0|0|1|1
7|#_SUBSET_GLOBAL|2|0|0|1|1
7|#_SUBSET_GLOBAL|2|0|0|1|1
8|#_VIEW_LOCAL|3|0|0|1|1
8|#_VIEW_LOCAL|3|0|0|1|1
9|#_VIEW_GLOBAL|2|0|0|1|1
9|#_VIEW_GLOBAL|2|0|0|1|1
11|#_GROUP_DIMENSION_DATA_Products|2|1352|0|1|1
13|#_GROUP_DIMENSION_DATA_Customers|2|1864|0|1|1
```

Pour plus d'informations concernant l'interprétation des informations sur le cube, consultez *Découvrir le schéma de sortie en lecture seule du tPaloCubeList, page 49*.

tPaloDatabase

Propriétés du tPaloDatabase

Famille de composant	Business Intelligence/Cube OLAP/Palo	
Fonction	Le composant **tPaloDatabase** crée, supprime ou recrée des bases de données sur un serveur Palo donné.	
Objectif	Ce composant gère les bases de données dans un serveur Palo.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
Connection configuration Indisponible lors de l'utilisation d'une connexion existante.	*Host Name*	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Server Port	Saisissez le numéro du port d'écoute du serveur Palo.
	Username et *Password*	Informations d'authentification de l'utilisateur de Palo.
	Database	Saisissez le nom de la base de données dans laquelle l'opération doit s'effectuer.

	Action on database	Sélectionnez l'opération que vous souhaitez effectuer sur le base de données: - **Create database** : la base de données n'existe pas et sera créée. - **Create database if not exists** : la base est créée lorsqu'elle n'existe pas. - **Delete database if exists and create** : la base de données est supprimée si elle existe et une nouvelle base de données est créée. - **Delete database** : la base de données est supprimée du serveur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en standalone lors d'opérations de gestion de bases de données dans un serveur Palo.	
Global Variables		**Databasename** : Indique le nom de la base de données traitée. Disponible en tant que variable **After**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* du **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	n/a	

Scénario : Créer une base de données

Le Job de ce scénario crée une nouvelle base de données sur un serveur Palo donné.

Pour reproduire ce scénario, procédez comme suit :

- Déposez un composant **tPaloDatabase** de la **Palette** dans l'espace de modélisation graphique.
- Double-cliquez sur le composant **tPaloDatabase** afin d'ouvrir sa vue **Component**.

- Dans le champ **Host name**, saisissez le nom de l'hôte ou l'adresse IP du serveur hôte, *localhost* dans cet exemple.
- Dans le champ **Server Port**, saisissez le numéro du port d'écoute du serveur Palo. Dans ce scénario, le numéro est *7777*.
- Dans les champs **Username** (identifiant) et **Password** (mot de passe), saisissez vos informations d'authentification. Dans cet exemple, les deux sont *admin*.
- Dans le champ **Database**, saisissez le nom de la base de données dans laquelle vous souhaitez créer le cube, *talenddatabase* dans cet exemple.
- Dans le champ **Action on database**, sélectionnez l'action à effectuer. Dans ce scénario, sélectionnez **Create database** puisque la base de données à créer n'existe pas.
- Appuyez sur **F6** pour exécuter votre Job.

Une nouvelle base de données est créée sur le serveur Palo donné.

tPaloDatabaseList

Propriétés du tPaloDatabaseList

Famille de composant	Business Intelligence/Cube OLAP/Palo	
Fonction	Le composant **tPaloDatabaseList** récupère les informations d'une liste de bases de données dans un serveur Palo donné.	
Objectif	Ce composant liste les noms des bases de données, leur type, le nombre de cubes, le nombre de dimensions, le statut et l'ID des bases de données d'un serveur Palo donné.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
Connection configuration Indisponible lors de l'utilisation d'une connexion existante.	*Host Name*	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Server Port	Saisissez le numéro du port d'écoute du serveur Palo.
	Username et *Password*	Informations d'authentification de l'utilisateur de Palo.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant peut être utilisé en tant que composant de début. Il nécessite un composant de sortie.	
Global Variables		**Number of databases** : Indique le nombre de bases de données traitées. Disponible en tant que variable **After**. Retourne un entier. **Database_id** : Indique l'ID de la base de données traitée. Disponible en tant que variable **Flow**. Retourne un long. **Databasename** : Indique le nom de la base de données traitée. Disponible en tant que variable **After**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* du **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	Le schéma de sortie est fixe et en lecture seule.	

Découvrir le schéma de sortie en lecture seule du tPaloDatabaseList

Le tableau ci-dessous présente les informations relatives au schéma de sortie en lecture seule du composant **tPaloDatabaseList**.

Base de données	Type	Description
Database_id	long	ID interne de la base de données.
Database_name	string	Nom de la base de données.
Database_dimensions	int	Nombre de dimensions dans la base de données.
Database_cubes	int	Nombre de cubes dans la base du données.
Database_status	int	Statut de la base de données . - **0** = non chargée - **1** = chargée - **2** = changée
Database_types	int	Type de la base de données. - **0** = normal - **1** = system - **3** = user info

Scénario : Récupérer des informations détaillées concernant les bases de données d'un serveur Palo donné

Le Job de ce scénario récupère des informations concernant toutes les bases de données d'un serveur Palo donné.

Pour reproduire ce scénario, procédez comme suit :

- Déposez un composant **tPaloDatabaseList** et un **tLogRow** de la **Palette** dans l'espace de modélisation graphique.
- Cliquez-droit sur le **tPaloDatabaseList** pour ouvrir le menu contextuel.
- Dans ce menu, sélectionnez **Row** > **Main** pour relier les deux composants.
- Double-cliquez sur le **tPaloDatabaseList** pour ouvrir sa vue **Component**.

- Dans le champ **Host name**, saisissez le nom de l'hôte ou l'adresse IP du serveur hôte, *localhost* dans cet exemple.
- Dans le champ **Server Port**, saisissez le numéro du port d'écoute du serveur Palo. Dans ce scénario, le numéro est *7777*.
- Dans les champs **Username** (identifiant) et **Password** (mot de passe), saisissez vos informations d'authentification. Dans cet exemple, les deux sont *admin*.
- Appuyez sur **F6** pour exécuter le Job.

Les informations de toutes les bases de données du serveur Palo sont récupérées et affichées dans la console de la vue **Run**.

```
0|System|7|6|1|1
1|Demo|23|23|1|0
6|Biker|27|28|1|0
9|tx_g|17|17|1|0
25|DBHSE24|13|12|1|0
31|elmCreate|13|12|1|0
32|tPaloOutputMulit|15|15|1|0
```

Pour plus d'informations concernant le schéma de sortie, consultez la section *Découvrir le schéma de sortie en lecture seule du tPaloDatabaseList, page 56*.

tPaloDimension

Propriétés du tPaloDimension

Famille de composant	Business Intelligence/Cube OLAP/Palo	
Fonction	Le composant **tPaloDimension** crée, supprime ou recrée des dimensions avec ou sans éléments de dimension dans une base de données Palo.	
Objectif	Ce composant gère les dimensions Palo, ainsi que les éléments dans une base de données.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
Connection configuration Indisponible lors de l'utilisation d'une connexion existante.	*Host Name*	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Server Port	Saisissez le numéro du port d'écoute du serveur Palo.
	Username et *Password*	Informations d'authentification de l'utilisateur de Palo.
	Database	Saisissez le nom de la base de données dans laquelle les dimensions sont gérées.
	Dimension	Saisissez le nom de la dimension sur laquelle l'opération donnée doit s'effectuer.

	Action on dimension	Sélectionnez l'opération que vous souhaitez effecteur sur la dimension : - **None** : aucune action n'est effectuée sur la dimension. - **Create dimension** : la dimension n'existe pas et sera créée. - **Create dimension if not exists** : cette dimension est créée quand elle n'existe pas. - **Delete dimension if exists and create** : cette dimension est supprimée si elle existe et une nouvelle dimension est créée. - **Delete dimension** : la dimension est supprimée de la base de données.
	Create dimension elements	Cochez cette case pour activer les champs de gestion des dimensions et créer des éléments de dimension , et créer cette dimension.
Les champs ci-dessous sont disponibles uniquement quand la case **Create dimension elements** est cochée.	*Dimension type* Disponible uniquement lorsque l'action à effectuer sur la dimension est **None**.	Sélectionnez le type de dimension à créer. Le type peut être : - **Normal** - **User info** - **System** - **Attribute**
	Commit size	Saisissez le nombre d'éléments qui seront créés avant de les sauvegarder dans la dimension.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.
	Consolidation type - None Cette option active les champs des paramètres correspondants devant être renseignés.	Sélectionnez ce bouton pour déplacer directement les éléments entrants dans la dimension donnée. Avec cette option, vous ne définissez auncune consolidation ou hiérarchie.
		Input Column : sélectionnez une colonne dans la liste déroulante. Les colonnes de la liste déroulante sont celles que vous avez définies dans le schéma. Les valeurs de la colone sélectionnée seront prises en compte lors du traitement d'éléments de dimension.

		Element type : Sélectionnez le type d'éléments. Il peut être : - **Numeric** - **Text**
		Creation mode : Sélectionnez le mode de création pour les éléments à traiter. Ce mode peut être : - **Add** : ajoute simplement un élément à la dimension. - **Force add** : force la création de cet élément. S'il existe, l'élément sera recréé. - **Update** : met à jour l'élément s'il existe. - **Add or Update** : si l'élément n'existe pas, il sera créé. S'il existe, il sera mis à jour. Cette option est l'option par défaut. - **Delete** : supprime cet élément de la dimension.
	Consolidation type - Normal Cette option active les champs des paramètres correspondants devant être renseignés.	Sélectionnez ce bouton pour créer des éléments et les consolider dans la dimension donnée. Cette consolidation structure les éléments créés à différents niveaux.
		Input Column: sélectionnez une colonne dans la liste déroulante. Les colonnes de la liste déroulante sont celles que vous avez définies dans le schéma. Les valeurs de la colone sélectionnée seront prises en compte lors du traitement d'éléments de dimension.
		Element type : Sélectionnez le type d'éléments. Il peut être : - **Numeric** - **Text**
		Creation mode : Sélectionnez le mode de création pour les éléments à créer. Ce mode peut être : - **Add** : ajoute simplement un élément à la dimension. - **Force add** : force la création de cet élément. S'il existe, l'élément sera recréé. - **Update** : met à jour l'élément s'il existe. - **Add or Update** : si l'élément n'existe pas, il sera créé. S'il existe, il sera mis à jour. Cette option est l'option par défaut.
	Consolidation type - Self-referenced Cette option active les champs des paramètres correspondants devant être renseignés.	Sélectionnez ce bouton pour créer des éléments et les structurer à partir d'une relation parent-enfant. Le flux d'entrée est responsable du regroupement de la consolidation.
	Element's type	Sélectionnez le type d'éléments. Il peut être : - **Numeric** - **Text**

	Creation mode	Sélectionnez le mode de création pour les éléments à créer. Ce mode peut être : - **Add** : ajoute simplement un élément à la dimension. - **Force add** : force la création de cet élément. S'il existe, l'élément sera recréé. - **Update** : met à jour l'élément s'il existe. - **Add or Update** : si l'élément n'existe pas, il sera créé. S'il existe, il sera mis à jour. Cette option est l'option par défaut.
		Input Column : sélectionnez une colonne dans la liste déroulante. Les colonnes dans cette liste sont celles que vous avez définies dans le schéma. Les valeurs de la colone sélectionnée seront prises en compte lors du traitement d'éléments de dimension.
		Hierarchy Element : sélectionnez le type et la relation de cette colonne d'entrée pour la consolidation. - **Parent** : définissez la valeur d'entrée en tant qu'élément parent. - **Child** : lie la valeur d'entrée à la valeur parent et construit la consolidation. - **Factor** : définissez le facteur de cette consolidation.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en standalone ou comme composant de fin d'un processus.	
Global Variables		**Dimensionname** : Indique le nom de la dimension traitée. Disponible en tant que variable **After** . Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* du **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Iterate **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	La suppression d'éléments de dimension est possible uniquement avec le type de consolidation **None**. Seul le type de consolidation **Self-Referenced** permet le placement d'un facteur dans cette consolidation.	

Scénario : Créer une dimension avec des éléments

Le Job de ce scénario crée une dimension de date avec une hiérarchie d'éléments simple, composée de trois niveaux : *Year, Month, Date* (Année, mois, date).

Pour reproduire ce scénario, procédez comme suit :

- Déposez un composant **tPaloConnection**, un **tRowGenerator**, un **tMap**, et un **tPaloDimension** de la **Palette** dans l'espace de modélisation graphique.
- Cliquez-droit sur le **tPaloConnection** afin d'ouvrir le menu contextuel.
- Dans le menu, sélectionnez **Trigger** > **On Subjob Ok** pour le relier au **tRowGenerator**.
- Cliquez-droit sur le **tRowGenerator** afin d'ouvrir le menu contextuel.

  Le **tRowGenerator** est utilisé pour générer des lignes au hasard afin de simplifier ce processus. Dans un cas réel d'utilisation, vous pouvez utiliser un autre composant d'entrée afin de charger vos données.

- Dans le menu, sélectionnez **Row** > **Main** pour le relier au **tMap**.
- Cliquez-droit sur le **tMap** pour ouvrir le menu contextuel.
- Dans le menu, sélectionnez **Row** > ***New output*** pour le relier au **tPaloDimension**.
- Nommez ce lien *out1* dans la boîte de dialogue qui s'ouvre.
- Double-cliquez sur le composant **tPaloCube** afin d'ouvrir sa vue **Component**.

- Dans le champ **Host name**, saisissez le nom de l'hôte ou l'adresse IP du serveur hôte, *localhost* dans cet exemple.
- Dans le champ **Server Port**, saisissez le numéro du port d'écoute du serveur Palo. Dans ce scénario, le numéro est *7777*.
- Dans les champs **Username** (identifiant) et **Password** (mot de passe), saisissez les informations d'authentification. Dans cet exemple, les deux sont *admin*.
- Double-cliquez sur le **tRowGenerator** pour ouvrir son éditeur.

- Dans la partie supérieure de l'éditeur, cliquez sur le bouton **[+]** pour ajouter une colonne, et renommez-la *random_date* dans la colonne **Column**.
- Dans la ligne nouvellement ajoutée, sélectionnez **Date** pour la colonne **Type**, et **getRandomDate** dans la colonne **Functions**.
- Dans la vue **Function parameters** de la partie inférieure de l'éditeur, saisissez les nouvelles valeurs des dates minimum et maximum de la colonne **Value**. Dans cet exemple, le minimum est *2010-01-01*, le maximum est *2010-12-31*.
- Cliquez sur **OK** afin de valider vos modifications et fermer l'éditeur.
- Dans la boîte de dialogue qui s'ouvre, cliquez sur **OK** pour propager les modifications.
- Double-cliquez sur le **tMap** afin d'ouvrir son éditeur.

- Dans la vue **Schema editor** de la partie inférieure de la fenêtre, sous la table *out1*, cliquez sur le bouton **[+]** afin d'ajouter trois lignes.
- Dans la colonne **Column** de la table *out1* saisissez de nouveaux noms pour les trois lignes ajoutées, *Year*, *Month* et *Date* (année, mois et date). Ces lignes sont ajoutées automatiquement à la table *out1* dans la partie inférieure du **tMap editor**.
- Dans la table *out1* de la partie supérieure de l'éditeur, cliquez sur la colonne **Expression** de la ligne *Year*.
- Appuyez sur les touches **Ctrl+Espace** afin d'ouvrir la liste d'auto-complétion.
- Double-cliquez sur TalendDate.formatDate pour la sélectionner. L'expression pour obtenir la date s'affiche dans la ligne Year. L'expression est la suivante : `TalendDate.formatDate("yyyy-MM-dd HH:mm:ss",myDate)`.
- Remplacez l'expression par défaut par `TalendDate.formatDate("yyyy",row1.random_date)`.
- Répétez l'opération pour les lignes Month et Date pour ajouter l'expression par défaut et remplacez-la respectivement par `TalendDate.formatDate("MM",row1.random_date)`et par `TalendDate.formatDate("dd-MM-yyyy", row1.random_date)`.
- Cliquez sur **OK** afin de valider cette modification et acceptez la propagation des modifications en cliquant sur **OK** dans la boîte de dialogue qui s'ouvre.
- Dans l'espace de modélisation graphique, double-cliquez sur le **tPaloDimension** afin d'ouvrir sa vue **Component**.

- Cochez la case **Use an existing connection**. Le **tPaloConnection_1** s'affiche automatiquement dans le champ **Connection configuration**.
- Dans le champ **Database**, saisissez le nom de la base de données dans laquelle la nouvelle dimension est créée, *talendDatabase* dans cet exemple.
- Dans le champ **Dimension**, saisissez le nom que vous souhaitez utiliser pour la dimension à créer, *Date*, par exemple.
- Dans le champ **Action on dimension**, sélectionnez l'action à effectuer. Dans ce scénario, sélectionnez **Create dimension if not exist**.
- Sélectionnez la case **Create dimension elements**.
- Dans la zone **Consolidation Type**, sélectionnez le bouton **Normal**.
- Sous le tableau de la hiérarchie des éléments, cliquez trois fois sur le bouton **[+]** pour ajouter trois lignes au tableau.
- .Dans la colonne **Input column** du tableau, sélectionnez *Year* dans la liste déroulante de la première ligne, *Month* dans la deuxième et *Date* dans la troisième. Cela détermine le niveau des éléments des différentes colonnes du schéma d'entrée.
- Appuyez sur **F6** pour exécuter le Job.

Une nouvelle dimension a été créée dans la base de données Palo *talendDatabase*.

Modeller - LOCALHOST / talendDatabase
Database
Date
2010
04
26-04-2010
25-04-2010
30-04-2010
09-04-2010
15-04-2010
04-04-2010
02-04-2010
12-04-2010
23-04-2010
11
01
19-01-2010
17-01-2010
15-01-2010
29-01-2010
08-01-2010
12-01-2010
06-01-2010

tPaloDimensionList

Propriétés du tPaloDimensionList

Famille de composant	Business Intelligence/Cube OLAP/Palo	
Fonction	Le composant **tPaloDimensionList** récupère les informations d'une liste de dimensions de la base de données Palo déterminée.	
Objectif	Ce composant affiche le nom des dimensions, leur type, le nombre d'éléments de dimension, le maximum d'indentation des dimensions, le maximum de profondeur des dimensions, le maximum des niveaux des dimensions, l'ID des dimensions dans un serveur Palo donné.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
Connection configuration Indisponible lors de l'utilisation d'une connexion existante.	*Host Name*	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Server Port	Saisissez le numéro du port d'écoute du serveur Palo.
	Username et *Password*	Informations d'authentification de l'utilisateur de Palo.

	Database	Saisissez le nom de la base de données dans laquelle se trouvent les dimensions qui vous intéressent.
	Retrieve cube dimensions	Cochez cette case pour récupérer les informations des dimensions d'un cube existant.
	Cube Disponible lorsque vous cochez la case **Retrieve cube dimensions**.	Saisissez le nom du cube duquel les informations concernant les dimensions sont récupérées.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en standalone ou comme composant de début d'un processus.	
Global Variables		**Dimension name** : Indique le nom de la dimension traitée. Disponible en tant que variable **Flow**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* du **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : Run if, On Subjob Ok; On Subjob Error, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate. **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	Le schéma de sortie est fixe et en lecture seule.	

Découvrir le schéma de sortie en lecture seule du tPaloDimensionList

Le tableau ci-dessous présente les informations relatives au schéma de sortie en lecture seule du composant **tPaloDimensionList**.

Dimension	Type	Description
Dimension_id	long	ID interne de la dimension.
Dimension_name	string	Nom de la dimension.
Dimension_attribute_cube	string	Nom du cube des attributs.
Dimension_rights_cube	string	Nom du cube des droits.
Dimension_elements	int	Nombre d'éléments de dimension.
Dimension_max_level	int	Niveau maximal de la dimension.
Dimension_max_indent	int	Indentation maximum de la dimension.
Dimension_max_depth	int	Profondeur maximale de la dimension.
Dimension_type	int	Type de la dimension : - **0** = normal - **1** = system - **2** = attribute - **3** = user info

Scénario: Récupérer des informations détaillées concernant les dimensions d'une base de données déterminée

Le Job de ce scénario récupère les informations de toutes les dimensions d'une base de données déterminée.

Pour reproduire ce scénario, procédez comme suit :

- Déposez un composant **tPaloDimensionList** et un **tLogRow** de la **Palette** dans l'espace de modélisation graphique.
- Cliquez-droit sur le **tPaloDimensionList** pour ouvrir le menu contextuel.
- Dans le menu, sélectionnez **Row** > **Main** pour relier les deux composants.
- Double-cliquez sur le composant **tPaloDimensionList** pour ouvrir sa vue **Component**.

- Dans le champ **Host name**, saisissez le nom de l'hôte ou l'adresse ID du serveur hôte, *localhost* dans cet exemple.
- Dans le champ **Server Port**, saisissez le numéro du port d'écoute du serveur Palo. Dans ce scénario, le numéro est *7777*.
- Dans les champs **Username** (identifiant) et **Password** (mot de passe), saisissez vos informations d'authentification. Dans cet exemple, les deux sont *admin*.
- Dans le champ **Database**, saisissez le nom de la base de données contenant les dimensions qui vous intéressent, *Biker* dans cet exemple.
- Appuyez sur **F6** pour exécuter le Job.

Les informations concernant les dimensions de la base de données *Biker* sont récupérées et affichées dans la console de la vue **Run**.

```
[statistics] connected
11|Products||#_Products|#_GROUP_DIMENSION_DATA_Products|338|3|4|3|0
13|Customers||#_Customers|#_GROUP_DIMENSION_DATA_Customers|466|3|4|3|0
15|Orderlines||#_Orderlines|#_GROUP_DIMENSION_DATA_Orderlines|3890|1|2|1|0
17|Channels||#_Channels|#_GROUP_DIMENSION_DATA_Channels|4|1|2|1|0
19|Years||#_Years|#_GROUP_DIMENSION_DATA_Years|11|1|2|1|0
21|Months||#_Months|#_GROUP_DIMENSION_DATA_Months|35|18|18|18|0
23|Datatypes||#_Datatypes|#_GROUP_DIMENSION_DATA_Datatypes|8|1|2|1|0
25|Measures||#_Measures|#_GROUP_DIMENSION_DATA_Measures|6|1|2|1|0
0|#_USER_||||5|0|1|0|0
1|#_GROUP_||||4|0|1|0|0
2|#_CUBE_||||2|0|1|0|0
4|#_CONFIGURATION_||||2|0|1|0|0
5|#_DIMENSION_||||8|0|1|0|0
7|#_SUBSET_||#_#_SUBSET_||0|0|0|0|0
9|#_VIEW_||#_#_VIEW_||0|0|0|0|0
3|#_#_CUBE__||||0|0|0|0|0
6|#_#_DIMENSION__||||0|0|0|0|0
8|#_#_SUBSET__||||0|0|0|0|0
10|#_#_VIEW__||||0|0|0|0|0
12|#_Products_||||0|0|0|0|0
14|#_Customers_||||0|0|0|0|0
16|#_Orderlines_||||0|0|0|0|0
18|#_Channels_||||0|0|0|0|0
20|#_Years_||||0|0|0|0|0
```

Pour plus d'informations concernant le schéma de sortie, consultez la section *Découvrir le schéma de sortie en lecture seule du tPaloDimensionList, page 70*.

tPaloInputMulti

Propriétés du tPaloInputMulti

Famille de composant	Business Intelligence/Cube OLAP/Palo	
Fonction	Le composant **tPaloInputMulti** récupère des données (des éléments et des valeurs) d'un cube Palo.	
Objectif	Ce composant récupère les valeurs stockées ou calculées ainsi que les enregistrements à l'extérieur d'un cube.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
Connection configuration Indisponible lors de l'utilisation d'une connexion existante.	*Host Name*	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Server Port	Saisissez le numéro du port d'écoute du serveur Palo.
	Username et *Password*	Informations d'authentification de l'utilisateur de Palo.
	Database	Saisissez le nom de la base de données dans laquelle se trouvent les éléments qui vous intéressent.

	Cube	Saisissez le nom du cube dans lequel les éléments de dimension à récupérer sont stockés.
	Cube type	Sélectionnez le type de cube dans la liste déroulante pour le cube sur lequel l'opération donnée doit s'effectuer : - **Normal** : Type de cube "normal", type par défaut. - **Attribut** : un cube **Attribute** sera créé avec la création d'un cube "normal". - **System** : Type de cube "système". - **User Info** : un cube **User Info** sera créé/modifié par ce composant.
	Commit size	Saisissez le nombre de lignes de chaque lot à récupérer.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Les colonnes **MEASURE** et **TEXT** sont en lecture seule, mais vous pouvez ajouter d'autres colonnes.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.
	Cube Query	Remplissez ce tableau pour préciser les données que vous souhaitez récupérer.
		Column : les colonnes du schéma sont ajoutées automatiquement à cette colonne, une fois définies dans l'éditeur du schéma. Les colonnes du schéma sont utilisées pour stocker les éléments de dimensions récupérés.
		Dimensions : saisissez le nom de chaque dimension du cube duquel vous souhaitez récupérer les éléments de dimensions. *L'ordre des dimensions dans cette colonne doit être cohérent avec l'ordre dans le cube stockant ces dimensions.*
		Elements : saisissez les éléments de dimensions desquels récupérer les données. Si plusieurs éléments d'une dimension sont requis, séparez-les avec une virgule.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant requiert un composant de sortie.	

Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate. **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** Talend Open Studio.
Limitation	Selon l'architecture des systèmes OLAP, une seule valeur (texte ou numérique) peut être récupérée du cube. Les colonnes **MEASURE** et **TEXT** sont fixes et en lecture seule.	

Scénario : Récupérer des éléments de dimensions dans un cube donné

Le Job de ce scénario récupère plusieurs éléments de dimensions du cube de démo Palo *Sales*.

Pour reproduire ce scénario, procédez comme suit :

- Déposez un composant **tPaloInputMulti** et un **tLogRow** de la **Palette** dans l'espace de modélisation graphique.
- Cliquez-droit sur le composant **tPaloInputMulti** afin d'ouvrir son menu contextuel.
- Dans le menu, sélectionnez **Row** > **Main** afin de relier le **tPaloInputMulti** au **tLogRow**.
- Double-cliquez sur le **tPaloInputMulti** afin d'ouvrir sa vue **Component**.

- Dans le champ **Host name**, saisissez le nom de l'hôte ou l'adresse IP du serveur hôte, *localhost* dans cet exemple.
- Dans le champ **Server Port**, saisissez le numéro du port d'écoute du serveur Palo. Dans ce scénario, le numéro est *7777*.
- Dans les champs **Username** (identifiant) et **Password** (mot de passe), saisissez vos informations d'authentification. Dans cet exemple, l'identifiant et le mot de passe sont *admin*.
- Dans le champ **Database**, saisissez le nom de la base de données dans laquelle est stocké le cube.
- Dans le champ **Cube**, saisissez le nom du cube dans lequel sont stockées les dimensions qui vous intéressent Dans ce scénario, le cube est celui de démo, *Sales*.
- Dans le champ **Cube type**, sélectionnez le type du cube à créer. Choisissez **Normal**, ce qui signifie que le cube sera de type "Normal", le type par défaut.
- A côté du champ **Edit Schema**, cliquez sur le bouton **[...]** pour ouvrir l'éditeur de schéma.

- Dans l'éditeur du schéma, cliquez sur le bouton **[+]** pour ajouter les lignes nécessaires. Dans cet exemple, ajoutez les lignes correspondant à toutes les dimensions stockées dans le cube *Sales* : *Products*, *Regions*, *Months*, *Years*, *Datatypes*, *Measures*. Saisissez les noms dans l'ordre donné de ce cube.
- Cliquez sur **OK** pour valider les modifications et acceptez la propagation du changement au composant suivant. Ces colonnes sont ajoutées automatiquement dans la colonne **Column** du tableau **Cube Query** dans la vue **Component**. Si l'ordre n'est pas cohérent par rapport à celui du cube *Sales*, déplacez les colonnes à l'aide des flèches montante et descendante.
- Dans la colonne **Dimensions** du tableau **Cube Query**, saisissez le nom de chaque dimension stockée dans le cube *Sales* selon l'ordre des lignes dans la colonne **Column**. Dans le cube *Sales*, le nom des dimensions est : *Products*, *Regions*, *Months*, *Years*, *Datatypes*, *Measures*.
- Dans la colonne **Elements** du tableau **Cube Query**, saisissez le nom des éléments de dimensions que vous souhaitez récupérer selon la dimension à laquelle ils appartiennent. Dans cet exemple, les éléments à récupérer sont *All products*, *Germany*, *Austria*, *Jan*, *2009*, *Actual*, *Turnover* (*Germany* et *Austria* appartenant à la même dimension **Regions**, ils doivent être saisis dans la même ligne et séparés par une virgule).
- Cliquez sur le **tLogRow** afin d'ouvrir sa vue **Component**.

- Dans la zone **Mode**, sélectionnez l'option **Table (print values in cell of a table)** pour afficher les résultats de l'exécution sous forme de tableau.
- Appuyez sur **F6** pour exécuter le Job.

Les éléments de dimensions et la valeur **Measure** correspondant s'affichent dans la console de la vue **Run**.

```
Starting job retrieve at 13:49 02/11/2010.

[statistics] connecting to socket on port 3338
[statistics] connected
.------------+--------+-------+-----+----------+---------+----------+----.
|                                 tLogRow_1                               |
|=-----------+--------+-------+-----+----------+---------+----------+---=|
|Products    |Regions|Months|Years|Datatypes|Measures|MEASURE   |TEXT|
|=-----------+--------+-------+-----+----------+---------+----------+---=|
|All Products|Austria|Jan   |2009 |Actual    |Turnover|476977.95|null|
|All Products|Germany|Jan   |2009 |Actual    |Turnover|1234.56   |null|
'------------+--------+-------+-----+----------+---------+----------+----'
[statistics] disconnected
Job retrieve ended at 13:49 02/11/2010. [exit code=0]
```

tPaloOutput

Propriétés du tPaloOutput

Famille de composant	Business Intelligence/Cube OLAP/Palo	
Fonction	Le composant **tPaloOutput** écrit une ligne de données (des éléments ainsi que des valeurs) dans un cube Palo donné.	
Objectif	Ce composant prend le flux d'entrée et l'écrit dans un Cube Palo donné.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
Connection configuration Indisponible lors de l'utilisation d'une connexion existante.	*Host Name*	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Server Port	Saisissez le numéro du port d'écoute du serveur Palo.
	Username et *Password*	Informations d'authentification de l'utilisateur de Palo.
	Database	Saisissez le nom de la base de données dans laquelle se trouve le cube qui vous intéresse.

	Cube	Saisissez le nom du cube dans lequel les données entrantes sont écrites.
	Commit size	Saisissez le nombre de lignes de chaque lot devant être écrit dans le cube.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.
	Column as Measure	Sélectionnez la colonne du flux d'entrée contenant les valeurs **Measure** ou **Text**.
	Create element if not exist	Cochez cette case afin de créer l'élément à traiter s'il n'existe pas.
	Save cube at process end	Cochez cette case afin de sauvegarder le cube dans lequel vous avez écrit les données à la fin du processus.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant requiert un composant d'entrée.	
Global variable		**Number of lines** : Indique le nombre de lignes traitées. Disponible en tant que variable **After**. Retourne un entier.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Iterate. **Trigger** : Run if. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	Ce composant ne peut écrire qu'une seule ligne de données dans un cube.	

Scénario associé

Pour un scénario associé, consultez le *Scénario 1 : Ecrire des données dans un cube donné, page 82*.

tPaloOutputMulti

Propriétés du tPaloOutputMulti

Famille de composant	Business Intelligence/Cube OLAP/Palo	
Fonction	Le composant **tPaloOutputMulti** écrit des données (des éléments et des valeurs) dans un cube Palo.	
Objectif	Ce composant prend la flux d'entrée et l'écrit dans un cube Palo donné.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
Connection configuration Indisponible lors de l'utilisation d'une connexion existante.	*Host Name*	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Server Port	Saisissez le numéro du port d'écoute du serveur Palo.
	Username et *Password*	Informations d'authentification de l'utilisateur de Palo.
	Database	Saisissez le nom de la base de données dans laquelle se trouve le cube qui vous intéresse.
	Cube	mé dans lequel les données entrantes sont écrites.

	Cube type	Sélectionnez le type de cube dans la liste déroulante pour le cube sur lequel l'opération donnée doit s'effectuer : - **Normal** : Type de cube "normal", type par défaut. - **Attribut** : un cube **Attribute** sera créé avec la création d'un cube "normal". - **System** : Type de cube "système". - **User Info** : un cube **User Info** sera créé/modifié par ce composant.
	Commit size	Saisissez le nombre de lignes de chaque lot devant être écrit dans le cube.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.
	Measure value	Sélectionnez la colonne du flux d'entrée contenant les valeurs **Measure** ou **Text**.
	Splash mode	Sélectionnez le mode splash utilisé pour écrire des données dans un élément consolidé. Le mode peut être : - **Add** :écrit des valeurs dans les éléments sous-jacents. - **Default** : utilise le mode splash par défaut. - **Set** : définit simplement la valeur et effectue la distribution à partir d'autres valeurs. - **Disable** : n'applique aucun splash. Pour plus d'informations concernant les modes splash Palo, consultez le Guide Utilisateur de Palo.
	Add values	Cochez cette case pour ajouter des nouvelles valeurs aux valeurs actuelles. Si vous ne cochez pas cette case, les nouvelles valeurs écraseront les valeurs actuelles.
	Use eventprocessor	Cochez cette case pour appeler le serveur de supervision.
	Die on error	Cette case est décochée par défaut, ce qui vous permet de terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant requiert un composant de début.	

Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main **Trigger** : Run if, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** Talend Open Studio.
Limitation	Les mesures numériques doivent être de type Double ou String. Lorsque le type String est utilisé, saisissez entre guillements la valeur devant être traitée.	

Scénario 1 : Ecrire des données dans un cube donné

Le Job de ce scénario écrit de nouvelles valeurs dans le cube *Sales* donné en démo dans la base de données *Demo* installée avec Palo.

Pour reproduire ce scénario, procédez comme suit :

- Déposez un **tFixedFlowInput** et un **tPaloOutputMulti** de la **Palette** dans l'espace de modélisation graphique.
- Cliquez-droit sur le composant **tFixedFlowInput** afin d'ouvrir son menu contextuel.
- Dans ce menu, sélectionnez **Row** > **Main** pour relier ce composant au **tPaloOutputMulti**.
- Double-cliquez sur le **tFixedFlowInput** pour ouvrir sa vue **Component**.

- Cliquez sur le bouton **[...]** pour ouvrir l'éditeur de schéma.

- Dans l'éditeur de schéma, cliquez sept fois sur le bouton **[+]** afin d'ajouter sept lignes. Renommez-les respectivement *Products*, *Regions*, *Months*, *Years*, *Datatypes*, *Measures* et *Values*. L'ordre de ces lignes doit être cohérent avec celui des dimensions correspondantes dans le cube *Sales* et le type de la colonne *Value*, qui comprend la valeur de la mesure, est défini comme double/Double.
- Cliquez sur **OK** afin de valider la modification et acceptez la propagation proposée par la boîte de dialogue qui s'ouvre. Les libellés des colonnes du schéma s'affichent automatiquement dans le tableau **Value** sous la case **Use single table** de la zone **Mode**.
- Dans le tableau **Value**, saisissez la valeur de chaque ligne dans la colonne **Value**. Ces valeurs sont : *Desktop L*, *Germany*, *Jan*, *2009*, *Actual*, *turnover*, *1234.56*.

- Double-cliquez sur le composant **tPaloOutputMulti** afin d'ouvrir sa vue **Component**.

- Dans le champ **Server Port**, saisissez le numéro du port d'écoute du serveur Palo. Dans ce scénario, le numéro est *7777*.
- Dans les champs **Username** (identifiant) et **Password** (mot de passe), saisissez vos informations d'authentification. Dans cet exemple, les deux sont *admin*.
- Dans le champ **Database**, saisissez le nom de la base de données dans laquelle vous souhaitez créer le cube, *Biker* dans cet exemple.
- Dans le champ **Cube**, saisissez le nom que vous souhaitez utiliser pour le cube à créer, *bikerTalend*, par exemple.
- Dans le champ **Cube type**, sélectionnez le type **Normal** dans la liste déroulante. Cela signifie que que le cube à créer sera de type "Normal", le type par défaut.
- Dans le champ **Measure Value**, sélectionnez **Value**.
- Appuyez sur **F6** pour exécuter le Job.

Le flux d'entrée de données a été écrit dans le cube *Sales*.

	A
1	localhost/Demo
2	Sales
3	Desktop L
4	**Germany**
5	Jan
6	2009
7	Actual
8	Turnover
9	
10	1 234,56
11	

Scénario 2 : Refuser un flux de données d'entrée lorsque les éléments à écrire n'existent pas dans un cube donné

Le Job de ce scénario essaye d'écrire des données dans le cube *Sales* mais puisque les éléments concernés n'existent pas dans ce cube, le flux d'entrée est refusé.

Pour reproduire ce scénario, procédez comme suit :

- Déposez les composants **tFixedFlowInput**, **tPaloCheckElements**, **tPaloOutputMulti** et **tLogRow** de la **Palette** dans l'espace de modélisation graphique.
- Cliquez-droit sur le **tFixedFlowInput** afin d'ouvrir son menu contextuel.
- Dans ce menu, sélectionnez **Row** > **Main** afin de connecter ce composant au **tPaloCheckElements**.
- Double-cliquez sur le **tFixedFlowInput** afin d'ouvrir sa vue **Component**.

- Cliquez sur le bouton **[...]** afin d'ouvrir l'éditeur de schéma.

- Dans l'éditeur de schéma, cliquez sept fois sur le bouton **[+]** afin d'ajouter sept lignes que vous allez nommer respectivement *Products*, *Regions*, *Months*, *Years*, *Datatypes*, *Measures* et *Values*. L'ordre de ces lignes doit être cohérent avec celui des dimensions correspondantes dans le cube *Sales* et le type de la colonne *Value*, qui comprend la valeur de la mesure et qui est défini comme double/Double.
- Cliquez sur **OK** afin de valider la modification et acceptez la propagation proposée par la boîte de dialogue qui s'ouvre. Les libellés des colonnes du schéma s'affichent automatiquement dans le tableau **Value** sous la case **Use single table** de la zone **Mode**.

- Dans le tableau **Value**, saisissez la valeur de chaque ligne dans la colonne **Value**. Ces valeurs sont : *Desktop L, Germany, Jan, 2009, Actual, turnover, 1234.56*.
- Double-cliquez sur le composant **tPaloCheckElements** afin d'ouvrir sa vue **Component**.

- Dans le champ **Host name**, saisissez *localhost*.
- Dans le champ **Server Port**, saisissez le numéro du port d'écoute du serveur Palo. Dans ce scénario, le numéro est *7777*.
- Dans les champs **Username** (identifiant) et **Password** (mot de passe), saisissez vos informations d'authentification. Dans cet exemple, les deux sont *admin*.
- Dans le champ **Database**, saisissez le nom de la base de données dans laquelle vous souhaitez créer le cube, *Demo* dans cet exemple.
- Dans le champ **Cube**, saisissez le nom du cube dans lequel vous souhaitez écrire les données, *Sales*, par exemple.
- Dans le champ **On Element Error**, sélectionnez **Reject** dans la liste déroulante.
- Dans le tableau contenant les éléments en bas de la vue **Basic settings**, cliquez sur la cellule au croisement de la ligne **Value** et de la colonne **Element type**. Sélectionnez **Measure** dans la liste déroulante.
- Double-cliquez sur le **tPaloOutputMulti** pour afficher sa vue **Component**.

- Dans le champ **Server Port**, saisissez le numéro du port d'écoute du serveur Palo. Dans ce scénario, le numéro est *7777*.
- Dans les champs **Username** (identifiant) et **Password** (mot de passe), saisissez vos informations d'authentification. Dans cet exemple, les deux sont *admin*.
- Dans le champ **Database**, saisissez le nom de la base de données dans laquelle vous souhaitez créer le cube, *Demo* dans cet exemple.
- Dans le champ **Cube**, saisissez le nom du cube dans lequel vous souhaitez écrire les données, *Sales*, par exemple.
- Dans le champ **Cube type**, sélectionnez le type **Normal** dans la liste déroulante. Cela signifie que que le cube à créer sera de type "Normal", le type par défaut.
- Dans le champ **Measure Value**, sélectionnez **Value**.
- Appuyez sur **F6** pour exécuter votre Job.

Les données à écrire sont rejetées et affichées dans la console de la vue **Run**. Vous pouvez voir que le message d'erreur est *Smart Products*.

```
Starting job JobOutputExistingElements at 14:45 09/11/2010.

[statistics] connecting to socket on port 3407
[statistics] connected
.--------------+-------+------+-----+---------+--------+-------+----------------------
|                                    tLogRow_1
|=-------------+-------+------+-----+---------+--------+-------+----------------------
|Products      |Regions|Months|Years|Datatypes|Measures|Value  |errorMessage
|=-------------+-------+------+-----+---------+--------+-------+----------------------
|Smart Products|Germany|Jan   |2009 |Actual   |Turnover|1234.56|Products:Smart Products
'--------------+-------+------+-----+---------+--------+-------+----------------------

[statistics] disconnected
Job JobOutputExistingElements ended at 14:45 09/11/2010. [exit code=0]
```

tPaloRule

Propriétés du tPaloRule

Famille de composant	Business Intelligence/Cube OLAP/Palo	
Fonction	Le composant **tPaloRule** crée ou modifie des règles dans un cube Palo donné.	
Objectif	Ce composant vous permet de gérer des règes dans un cube Palo donné.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
Connection configuration Indisponible lors de l'utilisation d'une connexion existante.	*Host Name*	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Server Port	Saisissez le numéro du port d'écoute du serveur Palo.
	Username et *Password*	Informations d'authentification de l'utilisateur de Palo.
	Database	Saisissez le nom de la base de données dans laquelle se trouvent les dimensions appliquant les règles.
	Cube	Saisissez le nom du cube dans lequel les informations de dimensions doivent être retrouvées.

	Cube rules	Remplissez ce tableau afin d'effectuer différentes actions sur les règles spécifiques : - **Definition** : saisissez la règle à appliquer.
		- **External Id** : saisissez l'ID externe personnalisé.
		- **Comment** : saisissez un commentaire pour cette règle.
		- **Activated** : cochez cette case afin d'activer la règle.
		- **Action** : sélectionnez dans la liste déroulante l'action à effectuer : - **Create** : créer cette règle. - **Delete** : supprimer cette règle. - **Update** : mettre à jour cette règle.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en standalone dans la création dynamique de cubes, avec une liste définie de dimensions.	
Connections		Liens de sortie (d'un composant à un autre) : **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	La mise à jour et la suppresion d'une règle ne sont disponibles uniquement lorsque la règle a été créée avec un ID externe.	

Scénario : Créer une règle dans un cube donné

Le Job de ce scénario crée une règle appliquée aux dimensions d'un cube donné.

Pour reproduire ce scénario, procédez comme suit :

- Déposez un composant **tPaloRule** de la **Palette** dans l'espace de modélisation graphique.
- Double-cliquez sur le composant **tPaloRule** afin d'ouvrir sa vue **Component**.

- Dans le champ **Host name**, saisissez le nom de l'hôte ou l'adresse IP du serveur hôte, *localhost* dans cet exemple.
- Dans le champ **Server Port**, saisissez le numéro du port d'écoute du serveur Palo. Dans ce scénario, le numéro est *7777*.
- Dans les champs **Username** (identifiant) et **Password** (mot de passe), saisissez vos informations d'authentification. Dans cet exemple, les deux sont *admin*.
- Dans le champ **Database**, saisissez le nom de la base de données dans laquelle se trouvent les dimensions appliquant les règles, *Biker* dans cet exemple.
- Dans le champ **Cube**, saisissez le nom du cube auquel appartiennent les dimensions appliquant les règles, *Orders* dans cet exemple.
- Sous le tableau **Cube rules**, cliquez sur le bouton **[+]** afin d'ajouter une ligne.
- Dans le tableau **Cube rules**, saisissez `['2009'] = 123` dans la colonne **Definition**, `OrderRule1` dans la colonne **External Id** et `Palo Demo Rules` dans la colonne **Comment**.
- Dans la colonne **Activated**, cochez la case.
- Dans la colonne **Action**, sélectionnez **Create** dans la liste déroulante.
- Appuyez sur **F6** afin d'exécuter le Job.

La nouvelle règle a été créée et la valeur de chaque élément de 2009 est *123*.

	A
1	localhost/Biker
2	Orders
3	**All Customers**
4	**All Channels**
5	**All Orders**
6	**All Datatypes**
7	Units
8	**All Products**
9	**All Months**
10	2009
11	
12	123
13	

tPaloRuleList

Propriétés du tPaloRuleList

Famille de composant	Business Intelligence/Cube OLAP/Palo	
Fonction	Le composant **tPaloRuleList** récupère les informations d'une liste de règles d'une base de données Palo spécifiée.	
Objectif	Ce composant liste toutes les règles, les formules, les commentaires, les statuts d'activation, les IDs externes d'un cube donné.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
Connection configuration Indisponible lors de l'utilisation d'une connexion existante.	*Host Name*	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Server Port	Saisissez le numéro du port d'écoute du serveur Palo.
	Username et *Password*	Informations d'authentification de l'utilisateur de Palo.
	Database	Saisissez le nom de la base de données dans laquelle se trouve le cube qui vous intéresse.

	Cube	Saisissez le nom du cube dans lequel les informations de règles doivent être retrouvées.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en standalone ou en tant que composant de début d'un processus.	
Global Variables		**Number of rules** : indique le nombre de règles traitées. Disponible en tant que variable **After**. Retourne un entier. **External ruleID** : indique l'ID externe de la règle traitée. Disponible en tant que variable **Flow**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* du **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	Le schéma de sortie est fixe et en lecture seule.	

Découvrir le schéma de sortie en lecture seule du tPaloRuleList

Le tableau ci-dessous présente les informations relatives au schéma de sortie en lecture seule du composant **tPaloRuleList**.

Base de données	Type	Description
rule_identifier	long	Identifiant interne de la règle.
rule_definition	string	Nom de la dimension.

Base de données	Type	Description
Dimension_attribute_cube	string	Formule de la règle. Pour plus d'informations concernant cette règle, consultez le Guide Utilisateur Palo.
rule_extern_id	string	ID externe personnalisé.
rule_comment	string	Commentaire utilisateur de la règle.
rule_activated	boolean	Indique si la règle a été activée ou non.

Scénario : Récupérer des informations détaillées concernant des règles d'un cube donné

Le Job de ce scénario récupère les informations concernant les règles appliquées dans les dimensions d'un cube donné.

Pour reproduire ce scénario, procédez comme suit :

- Déposez un composant **tPaloRuleList** et un **tLogRow** de la **Palette** dans l'espace de modélisation graphique.
- Cliquez-droit sur le **tPaloRuleList** afin d'ouvrir son menu contextuel.
- Dans ce menu, sélectionnez **Row** > **Main** afin de lier les deux composants.
- Double-cliquez sur le **tPaloRuleList** pour ouvrir sa vue **Component**.

- Dans le champ **Host name**, saisissez le nom de l'hôte ou l'adresse IP du serveur hôte, *localhost* dans cet exemple.

- Dans le champ **Server Port**, saisissez le numéro du port d'écoute du serveur Palo, *7777* dans cet exemple.
- Dans les champs **Username** (identifiant) et **Password** (mot de passe), saisissez vos informations d'authentification. Dans cet exemple, les deux sont *admin*.
- Dans le champ **Database**, saisissez le nom de la base de données dans laquelle se trouvent les dimensions appliquant les règles qui vous intéressent, *Biker* dans cet exemple.
- Dans le champ **Cube**, saisissez le nom du cube auquel les règles qui vous intéressent appartiennent.
- Appuyez sur **F6** pour exécuter le Job.

Les informations de toutes les règles du cube *Orders* sont récupérées et affichées dans la console de la vue **Run**.

```
Starting job tpalo at 16:34 12/11/2010.

[statistics] connecting to socket on port 3743
[statistics] connected
.---------------+---------------+--------------+---------------+--------------.
|                                tLogRow_1                                     |
|=--------------+---------------+--------------+---------------+--------------=|
|rule_identifier|rule_definition|rule_extern_id|rule_comment   |rule_activated|
|=--------------+---------------+--------------+---------------+--------------=|
|1              |['2009'] = 123 |OrderRule1    |Palo Demo Rules|true          |
'---------------+---------------+--------------+---------------+--------------'
[statistics] disconnected
Job tpalo ended at 16:35 12/11/2010. [exit code=0]
```

Pour plus d'informations concernant le schéma de sortie, consultez la section *Découvrir le schéma de sortie en lecture seule du tPaloRuleList, page 94*.

tParAccelSCD

Propriétés du tParAccelSCD

Famille du composant	Databases/Business Intelligence/ParAccel	
Fonction	Le composant **tParAccelSCD** reflète et traque les modifications d'une table ParAccel SCD dédiée.	
Objectif	Le **tParAccelSCD** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table SCD dédiée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case si vous utilisez un **tParAccelConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.

	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur Talend Open Studio**.
	SCD Editor	L'éditeur SCD Editor permet de constuire et de configurer les données du flux de sortie vers la table Slowly Changing Dimension. Pour plus d'informations, consultez la section *Méthodologie de gestion du SCD, page 20*.
	Use memory saving Mode	Cochez cette case pour améliorer les performances du système.
	Die on error	Cette case est décochée par défaut, ce qui vous permet de terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
	Debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
Utilisation	Ce composant est un composant de sortie. Par conséquent, il requiert un composant et une connexion de type Row Main en entrée.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23*.

tPostgresPlusSCD

Propriétés du tPostgresPlusSCD

Famille de composant	Databases/Business Intelligence/Postgresplus Server	
Fonction	Le composant **tPostgresPlusSCD** reflète et traque les modifications d'une table SCD PostgresPlus dédiée.	
Objectif	Le **tPostgresPlusSCD** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table SCD dédiée.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Property type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma de la base de données.

	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur Talend Open Studio**.
	SCD Editor	L'éditeur SCD Editor permet de constuire et de configurer les données du flux de sortie vers la table Slowly Changing Dimension. Pour plus d'informations, consultez la section *Méthodologie de gestion du SCD, page 20*.
	Use memory saving Mode	Cochez cette case pour améliorer les performances du système.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
	Debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
Utilisation	Ce composant est un composant de sortie. Par conséquent, il requiert un composant et une connexion de type Row Main en entrée.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23*.

tPostgresPlusSCDELT

Propriétés du tPostgresPlusSCDELT

Famille du composant	Databases/Business Intelligence/PostgresPlus	
Fonction	Le composant **tPostgresPlusSCDELT** reflète et traque les modifications d'une table PostgresPlus SCD dédiée.	
Objectif	Le **tPostgresPlusSCDELT** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table PostgresPlus SCD dédiée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tPostgresPlusConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.

	Source table	Nom de la table contenant les données à filtrer.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrire à la fois pour que l'opération d'insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Surrogate Key	Sélectionnez dans la liste une colonne à utiliser comme clé de substitution.
	Creation	Sélectionnez la méthode à utiliser pour générer la clé de substitution.
	Source Keys	Sélectionnez une colonne ou plus à utiliser en tant que clé(s) pour assurer l'unicité des données entrantes.
	Use SCD Type 1 fields	Utilisez le type 1 si vous n'avez pas besoin de traquer les modifications, pour des corrections typographiques par exemple. Sélectionnez les colonnes du schéma qui servira de référence pour les modifications.

	Use SCD Type 2 fields	Utilisez le type 2 si vous avez besoin de traquer les modifications, pour garder une trace des mises à jour effectuées par exemple. Sélectionnez les colonnes du schéma qui servira de référence pour les modifications. **Start date** : Ajoute une colonne à votre schéma SCD pour déterminer la valeur de la date de départ. Vous pouvez sélectionner l'une des colonnes d'entrée du schéma comme date de départ (Start Date) dans la table SCD. **End Date** : Ajoute une colonne à votre schéma SCD pour déterminer la valeur de la date de fin pour le journal. Lorsque le journal est en mode actif, la colonne **End Date** a une valeur nulle ; pour éviter cela, vous pouvez sélectionner l'option **Fixed Year value** et saisir une année fictive. **Log Active Status** : Ajoute une colonne à votre schéma SCD pour renseigner les valeurs de statut **true** et **false** . Cette colonne permet de repérer facilement le journal actif. **Log versions** : Ajoute une colonne à votre schéma SCD pour renseigner le numéro de version du journal.
Advanced settings	*Debug mode*	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie et une connexion de type **Row Main**.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23*.

tPostgresqlSCD

Propriétés du tPostgresqlSCD

Famille de composant	Databases/Business Intelligence/Postgresql Server	
Fonction	Le composant **tPostgresqlSCD** reflète et traque les modifications d'une table SCD Postgresql dédiée.	
Objectif	Le **tPostgresqlSCD** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table SCD dédiée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
	Use an existing connection	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma de la base de données.

	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.
	SCD Editor	L'éditeur SCD Editor permet de constuire et de configurer les données du flux de sortie vers la table Slowly Changing Dimension. Pour plus d'informations, consultez la section *Méthodologie de gestion du SCD, page 20*.
	Use memory saving Mode	Cochez cette case pour améliorer les performances du système.
	Die on error	Cette case est décochée par défaut, ce qui vous permet de terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
	Debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
Utilisation	Ce composant est un composant de sortie. Par conséquent, il requiert un composant et une connexion de type Row Main en entrée.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23*.

tPostgresqlSCDELT

Propriétés du tPostgresqlSCDELT

Famille du composant	Databases/Business Intelligence/Postgresql	
Fonction	Le composant **tPostgresqlSCDELT** reflète et traque les modifications d'une table Postgresql SCD dédiée.	
Objectif	Le **tPostgresqlSCDELT** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table Postgresql SCD dédiée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tPostgresqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Source table	Nom de la table contenant les données à filtrer.

	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrire à la fois pour que l'opération d'insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Surrogate Key	Sélectionnez dans la liste une colonne à utiliser comme clé de substitution.
	Creation	Sélectionnez la méthode à utiliser pour générer la clé de substitution.
	Source Keys	Sélectionnez une colonne ou plus à utiliser en tant que clé(s) pour assurer l'unicité des données entrantes.
	Use SCD Type 1 fields	Utilisez le type 1 si vous n'avez pas besoin de traquer les modifications, pour des corrections typographiques par exemple. Sélectionnez les colonnes du schéma qui servira de référence pour les modifications.

	Use SCD Type 2 fields	Utilisez le type 2 si vous avez besoin de traquer les modifications, pour garder une trace des mises à jour effectuées par exemple. Sélectionnez les colonnes du schéma qui servira de référence pour les modifications. **Start date** : Ajoute une colonne à votre schéma SCD pour déterminer la valeur de la date de départ. Vous pouvez sélectionner l'une des colonnes d'entrée du schéma comme date de départ (Start Date) dans la table SCD. **End Date** : Ajoute une colonne à votre schéma SCD pour déterminer la valeur de la date de fin pour le journal. Lorsque le journal est en mode actif, la colonne **End Date** a une valeur nulle ; pour éviter cela, vous pouvez sélectionner l'option **Fixed Year value** et saisir une année fictive. **Log Active Status** : Ajoute une colonne à votre schéma SCD pour renseigner les valeurs de statut **true** et **false** . Cette colonne permet de repérer facilement le journal actif. **Log versions** : Ajoute une colonne à votre schéma SCD pour renseigner le numéro de version du journal.
Advanced settings	*Debug mode*	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie et une connexion de type **Row Main**.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23*.

tSPSSInput

Avant de pouvoir utiliser toutes les fonctionnalités des composants SPSS, assurez-vous de de suivre les étapes suivantes :
-Si vous avez déjà installé SPSS, ajoutez le chemin du répertoire SPSS, qui se présente comme suit : SET PATH=%PATH%;<DR>:\program\SPSS.
-Si vous n'avez pas encore installé SPSS, copiez le lib SPSS IO "spssio32.dll" à partir du CD d'installation et copiez-le dans votre répertoire "system32".

Propriétés du tSPSSInput

Famille du composant	Business Intelligence	
Fonction	Le composant **tSPSSInput** lit les données de fichiers SPSS (.sav).	
Objectif	Le **tSPSSInput** envoie des données SPSS pour les écrire, dans un autre fichier par exemple.	
Basic settings	*Sync schema*	Cliquez sur ce bouton pour synchroniser le schéma avec les colonnes du fichier SPSS d'entrée.
	Schema et *Edit Schema*	Pour ce composant, le schéma des métadonnées est directement récupéré à partir du fichier SPSS d'entrée, c'est pourquoi il est en lecture seule. Vous pouvez cliquer sur **Edit schema** pour voir les métadonnées qui ont été récupérées.
	File name	Indiquez le nom ou le chemin d'accès au fichier SPPS à lire.
	Translate labels	Cochez cette case pour traduire les étiquettes (labels) des valeurs stockées. Si vous cochez cette case, vous devrez à nouveau récupérer les métadonnées.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé comme composant de début. Il requiert un composant de sortie.	

Scénario : Afficher le contenu d'un fichier SPSS

Le scénario suivant crée un Job à deux composants qui permet de lire chaque ligne d'un fichier SPSS (.sav) et de les afficher en sortie dans la console de log de la vue **Run**.

- A partir de la **Palette**, déposez les composants **tSPSSInput** et **tLogRow** dans l'espace graphique.

- Cliquez-droit sur le composant **tPSSInput** et connectez-le au **tLogRow** à l'aide d'un lien de type **Main Row**.
- Double-cliquez sur le composant **tSPSSInput** pour afficher la vue **Basic settings** et configurer ses paramètres.

- Cliquez sur le bouton **[...]** à côté du champ **Filename** pour parcourir vos dossiers jusqu'au fichier SPSS que vous voulez lire.
- Cliquez sur le bouton **[...]** à côté du champ **Sync schema**. Un message apparaît à l'écran pour vous inviter à récupérer le schéma du fichier SPSS sélectionné.

- Cliquez sur le bouton **Yes** pour fermer le message.
- Si nécessaire, cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour visualiser la structure de données prédéfinie dans le fichier SPSS source.

- Cliquez sur le bouton **OK** pour fermer la boîte de dialogue.
- Sauvegardez votre Job, et appuyez sur **F6** afin de l'exécuter.

Le fichier SPSS est lu ligne par ligne, et les champs extraits sont affichés dans la console.

```
Starting job SPSS at 10:28 20/01/2010.

[statistics] connecting to socket on port 3802
[statistics] connected
1.0|1.0|57997.0|198522.0
2.0|1.0|57113.0|203200.0
3.0|1.0|54123.0|200744.0
1.0|0.0|63936.0|187791.0
2.0|0.0|64835.0|195714.0
3.0|0.0|66804.0|208239.0
[statistics] disconnected
Job SPSS ended at 10:28 20/01/2010. [exit code=0]
```

Afin d'effectuer des traductions sur les valeurs stockées, procédez comme suit :

- Dans la vue **Basic settings**, cochez la case **Translate label** si vous voulez traduire les étiquettes (labels) des valeurs stockées.
- Cliquez à nouveau sur le bouton **Sync Schema**, afin de récupérer le schéma après traduction. Un message apparaît, vous demandant si vous souhaitez récupérer le schéma du fichier SPSS défini.
- Cliquez sur **Yes** (Oui) pour fermer le message et procéder à l'étape suivante.
  Un second message s'ouvre et vous demande si vous souhaitez propager les modifications.
- Cliquez sur **Yes** (Oui) afin de fermer le message et passer à l'étape suivante.
- Enregistrez le Job puis appuyez sur la touche **F6** pour l'exécuter.

Le fichier SPSS est lu ligne par ligne et les champs extraits sont affichés sur la console de log de la vue **Run**, après traduction des valeurs stockées.

```
Starting job SPSS at 10:32 20/01/2010.

[statistics] connecting to socket on port 3418
[statistics] connected
Under 21|Female|57997.0|198522.0
21-25|Female|57113.0|203200.0
26-30|Female|54123.0|200744.0
Under 21|Male|63936.0|187791.0
21-25|Male|64835.0|195714.0
26-30|Male|66804.0|208239.0
[statistics] disconnected
Job SPSS ended at 10:32 20/01/2010. [exit code=0]
```

tSPSSOutput

Avant de pouvoir utiliser toutes les fonctionnalités des composants SPSS, assurez-vous de de suivre les étapes suivantes :
-Si vous avez déjà installé SPSS, ajoutez le chemin du répertoire SPSS, qui se présente comme suit : SET PATH=%PATH%;<DR>:\program\SPSS.
-Si vous n'avez pas encore installé SPSS, copiez le lib SPSS IO "spssio32.dll" à partir du CD d'installation et copiez-le dans votre répertoire "system32".

Propriétés du tSPSSOutput

Famille du composant	Business Intelligence	
Fonction	Le composant **tSPSSOutput** écrit des données dans un fichier SPSS (.sav).	
Objectif	Le **tSPSSOutput** écrit ou ajoute des données à un fichier SPSS. Il crée les fichiers SPSS directement ou écrase ceux qui existent déjà.	
Basic settings	*Sync schema*	Cliquez sur ce bouton pour synchroniser le schéma avec les colonnes du fichier SPSS d'entrée.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Filename	Nom ou chemin d'accès du fichier SPSS à à écrire.
	Write Type	Sélectionnez dans la liste le type d'action à effectuer : **Write**: écrit simplement les nouvelles données. **Append**: écrit les nouvelles données à la suite de celles existantes.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant ne peut être utilisé comme composant de début. Il nécessite un flux d'entrée.	

Scénario : Ecrire des données dans un fichier .sav

Ce scénario décrit un Job très simple qui écrit des données d'entrée dans un fichier .sav.

- Glissez-déposez de la **Palette** dans l'espace de modélisation graphique un **tRowGenerator** et un **tSPSSOutput**.

- Cliquez-droit sur le **tRowGenerator**, et connectez-le au **tSPSSOutput**, à l'aide d'un lien **Row Main**.

- Dans l'espace de modélisation graphique, double-cliquez sur le **tRowGenerator** afin d'afficher sa vue **Basic settings** et ouvrir son éditeur. Vous pouvez définir votre schéma.

- Cliquez sur le bouton **[+]** afin d'ajouter les colonnes que vous voulez écrire dans le fichier .sav.
- Définissez le schéma, et configurez les paramètres des colonnes.

Vérifiez que vous avez défini la longueur de vos colonnes. Dans le cas contraire, un message d'erreur s'affichera lors de la contruction du Job.

- Cliquez sur **OK** afin de valider votre schéma et fermer l'éditeur.
- Double-cliquez sur le **tSPSSOutput** afin d'afficher sa vue **Component** et définir ses propriétés.

- Cliquez sur le bouton **[...]** à côté du champ **Filename**, et parcourez votre répertoire jusqu'au fichier SPSS .sav dans lequel vous souhaitez écrire les données.
- Cliquez sur le bouton **[...]** à côté du bouton **Sync columns**, afin de synchroniser les colonnes avec celles du composant précédent. Dans cet exemple, le schéma devant être inséré dans le fichier .sav est composé de deux colonnes : *id* et *country*.
- Si nécessaire, cliquez sur le bouton **Edit schema** pour voir/éditer le schéma défini.
- Dans la liste **Write Type**, sélectionnez **Write** (écrire) ou **Append** (écrire à la suite), pour simplement écrire les données d'entrée dans le fichier .sav, ou bien les écrire à la fin de ce même fichier.
- Sauvegardez votre Job, et appuyez sur **F6** pour l'exécuter.

Les données générées par le **tRowGenerator** sont écrites dans le fichier .sav défini.

tSPSSProperties

Pour pouvoir utiliser toutes les fonctionnalités des composants SPSS, assurez-vous de de suivre les étapes suivantes :
-Si vous avez déjà installé SPSS, ajoutez le chemin du répertoire SPSS, qui se présente comme suit : SET PATH=%PATH%;<DR>:\program\SPSS.
-Si vous n'avez pas encore installé SPSS, copiez le lib SPSS IO "spssio32.dll" à partir du CD d'installation et copiez-le dans votre répertoire "system32".

Propriétés du tSPSSProperties

Famille du composant	Business Intelligence	
Fonction	Le composant **tSPSSProperties** décrit les propriétés d'un fichier SPSS (.sav) défini.	
Objectif	Le **tSPSSProperties** permet d'obtenir des informations sur les propriétés principales d'un fichier SPSS défini.	
Basic settings	*Schema* et *Edit Schema*	Pour ce composant, le schéma des métadonnées est prédéfini, c'est pourquoi il est en lecture seule. Vous pouvez cliquer sur **Edit schema** pour voir les métadonnées prédéfinies. Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Filename	Nom ou chemin d'accès au fichier SPSS à traiter.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé comme composant de début. Il requiert un composant de sortie.	

Scénario associé

Pour des scénarios associés, consultez :

- *Scénario : Afficher le contenu d'un fichier SPSS, page 109*
- *Scénario : Ecrire des données dans un fichier .sav, page 113*

tSPSSStructure

Avant de pouvoir utiliser toutes les fonctionnalités des composants SPSS, assurez-vous de de suivre les étapes suivantes :
-Si vous avez déjà installé SPSS, ajoutez le chemin du répertoire SPSS, qui se présente comme suit : SET PATH=%PATH%;<DR>:\program\SPSS.
-Si vous n'avez pas encore installé SPSS, copiez le lib SPSS IO "spssio32.dll" à partir du CD d'installation et copiez-le dans votre répertoire "system32".

Propriétés du tSPSSStructure

Famille du composant	Business Intelligence	
Fonction	Le composant **tSPSSStructure** récupère les informations sur les variables contenues dans le fichier SPSS (.sav).	
Purpose	Le **tSPSSStructure** traite les variables contenues dans des fichiers SPSS. Vous pouvez utiliser ce composant avec le **tFileList** pour récupérer les informations des fichiers *.sav existant et les utiliser pour analyser de manière plus approfondie ou pour vérifier vos résultats.	
Basic settings	*Schema* et *Edit Schema*	Pour ce composant, le schéma des métadonnées est prédéfini, c'est pourquoi il est en lecture seule. Il correspond à la convention interne SPSS. Vous pouvez cliquer sur **Edit schema** pour voir les métadonnées prédéfinies. Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Filename	Nom ou chemin d'accès au fichier SPSS à traiter.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé comme composant de début. Il requiert un composant de sortie.	

Scénario associé

Pour des scénarios associés, consultez :

- *Scénario : Afficher le contenu d'un fichier SPSS, page 109*

- *Scénario : Ecrire des données dans un fichier .sav, page 113*

tSybaseSCD

Propriétés du tSybaseSCD

Famille du composant	Databases/Business Intelligence/Sybase	
Fonction	Le composant **tSybaseSCD** reflète et traque les modifications d'une table SCD Sybase dédiée.	
Objectif	**Le tSybaseSCD** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table SCD dédiée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.

	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur Talend Open Studio**.
	SCD Editor	L'éditeur SCD Editor permet de constuire et de configurer les données du flux de sortie vers la table Slowly Changing Dimension. Pour plus d'informations, consultez la section *Méthodologie de gestion du SCD, page 20*.
	Use memory saving Mode	Cochez cette case pour améliorer les performances du système.
	Die on error	Cette case est décochée par défaut, ce qui vous permet de terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
	Debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
Utilisation	Ce composant est un composant de sortie. Par conséquent, il requiert un composant et une connexion de type Row Main en entrée.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23*.

tSybaseSCDELT

Propriétés du tSybaseSCDELT

Famille du composant	Databases/Business Intelligence/Sybase	
Fonction	Le composant **tSybaseSCDELT** reflète et traque les modifications d'une table Sybase SCD dédiée.	
Objectif	Le **tSybaselSCDELT** adresse des besoins en transformation Slowly Changing Dimension, en lisant régulièrement une source de données et en répertoriant les modifications dans une table Sybase SCD dédiée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tSybaseConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Source table	Nom de la table contenant les données à filtrer.

	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrire à la fois pour que l'opération d'insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Surrogate Key	Sélectionnez dans la liste une colonne à utiliser comme clé de substitution.
	Creation	Sélectionnez la méthode à utiliser pour générer la clé de substitution.
	Source Key	Sélectionnez une colonne ou plus à utiliser en tant que clé(s) pour assurer l'unicité des données entrantes.
	Use SCD Type 1 fields	Utilisez le type 1 si vous n'avez pas besoin de traquer les modifications, pour des corrections typographiques par exemple. Sélectionnez les colonnes du schéma qui servira de référence pour les modifications.

	Use SCD Type 2 fields	Utilisez le type 2 si vous avez besoin de traquer les modifications, pour garder une trace des mises à jour effectuées par exemple. Sélectionnez les colonnes du schéma qui servira de référence pour les modifications. **Start date** : Ajoute une colonne à votre schéma SCD pour déterminer la valeur de la date de départ. Vous pouvez sélectionner l'une des colonnes d'entrée du schéma comme date de départ (Start Date) dans la table SCD. **End Date** : Ajoute une colonne à votre schéma SCD pour déterminer la valeur de la date de fin pour le journal. Lorsque le journal est en mode actif, la colonne **End Date** a une valeur nulle ; pour éviter cela, vous pouvez sélectionner l'option **Fixed Year value** et saisir une année fictive. **Log Active Status** : Ajoute une colonne à votre schéma SCD pour renseigner les valeurs de statut **true** et **false**. Cette colonne permet de repérer facilement le journal actif. **Log versions** : Ajoute une colonne à votre schéma SCD pour renseigner le numéro de version du journal.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie et une connexion de type **Row Main**.	

Scénario associé

Pour un scénario associé, consultez le composant *tMysqlSCD, page 19* et son *Scénario : Traquer des modifications avec les Slowly Changing Dimensions de type 0 à 3, page 23*.

Composants Custom Code

Ce chapitre passe en revue les principaux composants présents dans la famille **Custom Code** de la Palette de **Talend Open Studio**.

La famille **Custom Code** regroupe tous les composants répondant à un besoin spécifique de création de code à la volée.

tGroovy

Propriétés du tGroovy

Famille de composant	Custom Code	
Fonction	Le **tGroovy** permet de saisir du code personnalisé afin de l'intégrer dans le programme Talend. Ce code est exécuté une seule fois.	
Objectif	Le **tGroovy** permet d'étendre les fonctionnalités du Job Talend grâce au langage Groovy, qui est une syntaxe simplifiée du Java..	
Basic settings	*Groovy Script*	Saisissez le code Groovy que vous souhaitez exécuter.
	Variables	Ce tableau comprend deux colonnes : **Name** : nom de la variable appelée dans le code. **Value** : valeur associée à cette variable.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en standalone, ou en tant que sous-job à un seul composant.	
Limitation	Il est nécessaire de connaître le langage Groovy.	

Scénarios associés

- Pour un scénario utilisant du Code Groovy, consultez le *Scénario : Appeler du code Groovy contenu dans un fichier, page 127*.
- Pour un exemple d'utilisation proche dans le mode de fonctionnement, consultez le *Scénario : Imprimer le contenu d'une variable, page 129* du composant **tJava**.

tGroovyFile

Propriétés du tGroovyFile

Famille de composant	Custom Code	
Fonction	Le **tGroovyFile** permet d'appeler un script existant développé en Groovy.	
Objectif	Le **tGroovyFile** permet d'étendre les fonctionnalités du Job Talend grâce au langage Groovy, qui est une syntaxe simplifiée du Java..	
Basic settings	*Groovy File*	Nom et chemin d'accès au fichier contenant votre code Groovy.
	Variables	Ce tableau comprend deux colonnes : **Name** : nom de la variable appelée dans le code. **Value** : valeur associée à cette variable.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en standalone, ou en tant que sous-job à un seul composant.	
Limitation	Il est nécessaire de connaître le langage Groovy.	

Scénario : Appeler du code Groovy contenu dans un fichier

Ce scénario comprend un composant unique, le **tGroovyFile**, utilisé en standalone. Le Job a pour but d'appeler un fichier contenant du code Groovy, afin de retourner dans la **Console** les informations contenues dans ce fichier. Il se présente comme suit :

- Dans la **Palette**, ouvrez le dossier **Custom_Code**, puis glissez-déposez un composant **tGroovyFile** dans l'espace de modélisation graphique.
- Double-cliquez sur le composant afin d'afficher sa vue **Component**.

- Dans le champ **Groovy File**, saisissez le chemin d'accès au fichier contenant votre Code Groovy, ou parcourez votre répertoire.
- Dans le tableau **Variables**, ajoutez une ligne, en appuyant sur le bouton **[+]**.
- Dans la colonne **Name**, saisissez *"age"*, puis, dans, la colonne **Value**, saisissez *50*, comme dans la capture d'écran.

- Appuyez sur **F6** pour sauvegarder et exécuter votre Job.
- La **Console** affiche les informations contenues dans le fichier d'entrée, auxquelles s'ajoute le résultat de la variable.

```
Starting job tGroovyFile at 09:49 17/02/2010.

[statistics] connecting to socket on port 4016
[statistics] connected
The captain is 50 years old
[statistics] disconnected
Job tGroovyFile ended at 09:49 17/02/2010. [exit code=0]
```

tJava

Propriétés du tJava

Famille de composant	Custom Code	
Fonction	Le **tJava** permet de saisir du code personnalisé afin de l'intégrer dans le programme Talend. Ce code est exécuté une seule fois.	
Objectif	Le **tJava** permet d'étendre les fonctionnalités du Job Talend grâce au langage Java.	
Basic settings	*Code*	Saisissez le code Java que vous souhaitez exécuter. Pour plus d'informations concernant la syntaxe des fonctions Java spécifiques à Talend, consultez l'aide en ligne de **Talend Open Studio** dans le menu Help Contents > Developer Guide > API Reference. Pour la référence Java complète, allez sur : http://java.sun.com/javaee/6/docs/api/
Advanced settings	*Import*	Saisissez le code Java permettant d'importer, si nécessaire, la librairie externe utilisée dans le champ **Main code** de l'onglet **Basic settings**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé en tant que sous-job à un seul composant.	
Limitation	Il est nécessaire de connaître le langage Java.	

Scénario : Imprimer le contenu d'une variable

Le scénario suivant est une simple démonstration de l'application possible du composant tJava. L'objectif de ce Job est d'imprimer les lignes traitées à l'aide d'une commande Java et des variables globales fournies dans **Talend Open Studio**.

- Cliquez et déposez les composants dans l'espace de modélisation : **tFileInputDelimited**, **tFileOutputExcel**, **tJava**.
- Connectez le composant **tFileInputDelimited** au **tFileOutputExcel** à l'aide d'une connexion de type **Row Main**. Le contenu du fichier TXT délimité sera transmis à un fichier de type XSL via cette connexion sans subir de transformation.
- Ensuite, connectez le composant **tFileInputDelimited** au composant **tJava** à l'aide d'un lien de type **Then Run**. Ce lien met en place un séquençage ordonnant au tJava de se lancer à la fin du processus principal.
- Définissez les paramètres du composant **tFileInputDelimited** dans l'onglet **Basic settings** de la vue **Component**. Le fichier d'entrée utilisé en exemple est un simple fichier texte à deux colonnes : Name (des noms de personnes) et leur Emails respectifs.

- Pour cet exemple, le schéma n'a pas été centralisé dans le référentiel, vous devez donc configurer manuellement le schéma de ces deux colonnes.
- Cliquez sur le bouton **Edit Schema**.

- Une boîte de dialogue vous demandant si vous souhaitez propager les modifications apportées apparaît, cliquez sur OK pour accepter la propagation. Ainsi, le schéma du composant **tFileOutputExcel** est automatiquement renseigné à partir du schéma du composant précédent. Vous n'aurez donc pas à définir le schéma de nouveau.
- Paramétrez le fichier de sortie afin qu'il reçoive le contenu du fichier d'entrée sans modifications. Si le fichier n'existe pas déjà, il sera créé.

- Dans cet exemple, dans le champ **Sheet name**, indiquez le nom de la feuille Excel, ici Email et la case **Header** est cochée pour indiquer que le fichier contient un en-tête.
- Puis sélectionnez le composant **tJava** pour définir la commande Java à exécuter.

- Dans la zone **Code**, saisissez la commande suivante :
```
String var = "Nb of line processed: ";
var = var + globalMap.get("tFileInputDelimited_1_NB_LINE");
System.out.println(var);
```
- Dans cet exemple précis, utilisez la variable **NB_Line**. Pour accéder à la liste des variables globales, appuyez sur **Ctrl** + **Espace** de votre clavier et sélectionnez la variable globale correspondante.
- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

```
Starting job JavaDb at 13:53 20/08/2007.
Nb of line processed: 4
Job JavaDb ended at 13:53 20/08/2007. [exit code=0]
```

Le contenu est donc transmis au fichier Excel spécifié et le nombre de lignes traitées est affiché dans la console **Run**.

tJavaFlex

Propriétés du tJavaFlex

Famille de composant	Custom Code	
Fonction	Le **tJavaFlex** permet de saisir du code personnalisé afin de l'intégrer dans le programme Talend. Avec le **tJavaFlex**, vous pouvez saisir les trois différentes parties du code Java (start, main et end) constituant un composant qui permet d'exécuter une opération souhaitée et non disponible dans les actuels composants **Talend**.	
Objectif	Le **tJavaFlex** permet d'étendre les fonctionnalités du Job **Talend** grâce au langage Java.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Cliquez sur **Edit schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Data Auto Propagate	Cochez cette case pour propager automatiquement les données au composant suivant. Lorsque vous cochez cette case, les données récupérées ne peuvent être transformées a posteriori dans le champ Main code
	Start code	Saisissez le code Java qui va être appelé lors de la phase d'initialisation.
	Main code	Saisissez le code Java qui va être appliqué pour chaque ligne du flux.
	End code	Saisissez le code Java qui va être appelé lors de la phase de clôture.
Advanced settings	*Import*	Saisissez le code Java permettant d'importer, si nécessaire, la librairie externe utilisée dans le champ **Main code** de l'onglet **Basic settings**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant peut être utilisé en tant que composant de début, de milieu et de fin de Job, ainsi qu'en tant que sous-job à un seul composant.
Limitation	Il est nécessaire de connaître le langage Java.

Scénario 1 : Générer un flux de données

Ce scénario décrit un Job à deux composants dont le but est de générer un flux de trois lignes décrivant les différentes civilité : mademoiselle (Miss), madame (Mrs) et monsieur (Mr), et de les afficher dans la console.

- Cliquez-glissez un composant **tJavaFlex** de la famille **Custom Code** et un composant **tLogRow** de la famille **Logs & Errors** dans le Job Designer.
- Connectez-les via un lien **Row Main**.
- Double-cliquez sur le composant **tJavaFlex** pour paramétrer ses propriétés.

- Dans la liste **Schema Type**, sélectionnez le mode **Built-in** et cliquez sur le bouton **[...]** à coté du champ **Edit schema** pour décrire manuellement la structure des données à passer au composant suivant.

- Cliquez sur le bouton **[+]** pour ajouter deux colonnes : *key* de type **Integer** et *value* de type **String**.
- Cliquez **Ok** pour valider les changements et fermer la boîte de dialogue.
- Dans les propriétés du **tJavaFlex**, cochez la case **Data Auto Propagate** pour propager automatiquement les données au composant suivant.
  Dans cet exemple, n'effectuez pas de transformation sur les données récupérées.
- Dans le champ **Start code**, saisissez le code qui sera exécuté lors de la phase d'initialisation.
  Ici, le code indique l'initialisation du composant **tJavaFlex** en affichant le message START et met en place la boucle et les variables utilisées par la suite dans le code Java :
  ```
  System.out.println("## START\n#");
  String [] valueArray = {"Miss", "Mrs", "Mr"};

  for (int i=0;i<valueArray.length;i++) {
  ```

Main code

```
// Main part of your Java code
(loop)

row1.key = i;
row1.value = valueArray[i];
```

- Dans le champ **Main code**, saisissez le code qui sera appliqué pour chaque ligne de données.
  Ici, l'objectif est de retourner chaque clé (*key*) avec sa valeur :
  ```
  row1.key = i;
  row1.value = valueArray[i];
  ```

Dans le champ **Main code**, **row1** *correspond au nom du lien qui sort du* **tJavaFlex**. *Si vous renommez ce lien, il faut penser à modifier le code de ce champ.*

End code

```
// End part of your Java code

}
System.out.println("#\n##
END");
```

- Dans le champ **End code**, saisissez le code qui sera exécuté lors de la phase de clôture. Ici, l'accolade ferme la boucle et le code indique la fin de l'exécution du **tJavaFlex** en affichant le message END :
```
}
System.out.println("#\n## END");
```
- Assurez-vous que le schéma a bien été propagé au **tLogRow**.
- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

```
Starting job tJavaFlex_scenario1 at 14:49 02/09/2009.

## START
#
0|Miss
1|Mrs
2|Mr
#
## END
Job tJavaFlex_scenario1 ended at 14:49 02/09/2009. [exit code=0]
```

La console affiche les trois civilités, ainsi que leur clés correspondantes.

Scénario 2 : Traiter des lignes de données avec le tJavaFlex

Ce scénario décrit un Job à deux composants générant des données de manière aléatoire, récupérant et traitant ces données ligne par ligne avec du code Java via le **tJavaFlex**.

- Cliquez-glissez un composant **tRowGenerator** de la famille **Misc** et un composant **tJavaFlex** de la famille **Custom Code** dans le Job Designer.
- Connectez-les via un lien **Row Main**.
- Double-cliquez sur le composant **tRowGenerator** pour ouvrir son éditeur.

- Dans l'éditeur, cliquez sur le bouton **[+]** pour ajouter quatre colonnes : *number*, *txt*, *date* et *flag*.
- Définissez le schéma en paramétrant ces quatres colonnes conformément à la capture d'écran précédente.
- Dans la colonne **Functions**, sélectionnez la fonction **[...]** dans la liste de chaque colonne.
- Dans la colonne **Parameters**, saisissez 10 paramètres différents pour chaque colonne. Ces 10 paramètres correspondent aux données qui seront retournées de manière aléatoire lors de l'exécution du **tRowGenerator**.
- Cliquez sur **OK**.
- Double-cliquez sur le composant **tJavaFlex** pour paramétrer ses propriétés.

- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent.
- Dans le champ **Start code**, saisissez le code qui sera exécuté lors de la phase d'initialisation. Ici, le code indique l'initialisation du composant **tJavaFlex** en affichant le message START et définit une variable qui sera utilisée par la suite dans le code Java :

```
System.out.println("## START\n#");
int i = 0;
```

Main code

```
System.out.print(" row" + i + ":");
System.out.print("# number:" +
 row1.number);
System.out.print (" | txt:" +
 row1.txt.toUpperCase());
System.out.print(" | date:" +
 row1.date);
if(row1.flag) System.out.println(" |
flag: true");
else  System.out.println(" | flag:
false");

i++;
```

- Dans le champ **Main code**, saisissez le code qui sera appliqué pour chaque ligne de données. Ici, l'objectif est d'indiquer le numéro de la ligne en partant de 0, puis afficher le numéro et le texte aléatoire que vous mettez en majuscules et la date aléatoire définis dans le **tRowGenerator**. Ensuite, créez une condition affichant si le status est **true** (vrai) ou **false** (faux) et incrémentez le numéro de la ligne :

```
System.out.print(" row" + i + ":");
System.out.print("# number:" + row1.number);
System.out.print (" | txt:" + row1.txt.toUpperCase());
System.out.print(" | date:" + row1.date);
if(row1.flag) System.out.println(" | flag: true");
else  System.out.println(" | flag: false");

i++;
```

Dans le champ **Main code**, **row1** *correspond au nom du lien qui sort du* **tJavaFlex**. *Si vous renommez ce lien, il faut penser à modifier le code de ce champ.*

End code

```
System.out.println("#\n## END");
```

- Dans le champ **End code**, saisissez le code qui sera exécuté lors de la phase de clôture. Ici, le code indique la fin de l'exécution du composant **tJavaFlex** en affichant le message END :

```
System.out.println("#\n## END");
```

- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job tJavaFlex_scenario2 at 18:35 02/09/2009.

## START
#
 row0:# number:10 | txt:TEXT5 | date:2006-05-25 | flag: false
 row1:# number:7 | txt:TEXT7 | date:2006-06-26 | flag: true
 row2:# number:5 | txt:TEXT2 | date:2006-05-25 | flag: false
 row3:# number:6 | txt:TEXT1 | date:2005-08-28 | flag: true
 row4:# number:8 | txt:TEXT9 | date:2006-05-25 | flag: false
 row5:# number:9 | txt:TEXT1 | date:2007-01-21 | flag: false
 row6:# number:7 | txt:TEXT5 | date:2004-11-21 | flag: true
 row7:# number:9 | txt:TEXT4 | date:2005-08-29 | flag: true
 row8:# number:4 | txt:TEXT6 | date:2006-06-26 | flag: false
 row9:# number:3 | txt:TEXT10 | date:2006-05-25 | flag: false
#
## END
Job tJavaFlex_scenario2 ended at 18:35 02/09/2009. [exit
code=0]
```

La console affiche les données générées aléatoirement et modifiées à l'aide du code Java dans le **tJavaFlex**.

tJavaRow

Propriétés du tJavaRow

Famille de composant	Custom Code	
Fonction	Le **tJavaRow** permet de saisir du code personnalisé afin de l'intégrer dans le programme Talend. Avec le **tJavaRow**, vous pouvez saisir le code Java qui va être appliqué pour chaque ligne du flux.	
Objectif	Le **tJavaRow** permet d'étendre les fonctionnalités du Job **Talend** grâce au langage Java.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Cliquez sur **Edit schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Code	Saisissez le code Java qui va être appliqué pour chaque ligne du flux.
Advanced settings	*Import*	Saisissez le code Java permettant d'importer, si nécessaire, la librairie externe utilisée dans le champ **Main code** de l'onglet **Basic settings**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé en tant que composant de milieu de Job, et doit être relié à un composant de début et à un composant de sortie.	
Limitation	Il est nécessaire de connaître le langage Java.	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tLibraryLoad

Propriétés du tLibraryLoad

Famille de composant	Custom Code	
Fonction	Le **tLibraryLoad** permet d'importer une librairie.	
Objectif	Le **tLibraryLoad** permet de charger des librairies Java utilisables dans un Job.	
Basic settings	*Library*	Sélectionnez dans la liste la librairie à importer, ou cliquez sur le bouton **[...]** afin de parcourir votre répertoire et choisir la librairie que vous souhaitez importer.
Advanced settings	*Dynamic Libs*	**Lib Paths** : Saisissez entre guillemets le chemin d'accès à votre librairie.
	Import	Saisissez le code Java permettant d'importer, si nécessaire, la librairie externe utilisée dans le champ **Main code** de l'onglet **Basic settings**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en standalone, mais il est plus logique de l'utiliser dans un Job.	
Limitation	n/a	

Scénario : Vérifier le format d'une adresse e-mail

Ce scénario comprend deux composants, un **tLibraryLoad** et un **tJava**. Le but de ce scénario est de vérifier le format d'une adresse e-mail et de retourner si elle est valide ou non.

- Dans la **Palette**, ouvrez le dossier **Custom_Code**, puis glissez-déposez les composants **tLibraryLoad** et **tJava** dans l'espace de modélisation graphique.

- Reliez le **tLibraryLoad** au **tJava** à l'aide d'un lien **Trigger** > **OnSubjobOk**.
- Double-cliquez sur le **tLibraryLoad** pour afficher sa vue **Basic settings**. Dans la liste **Library**, sélectionnez *jakarta-oro-2.0.8.jar*.
- Dans le champ **Import**, de l'onglet **Advanced settings**, saisissez *import org.apache.oro.text.regex.*;*

- Double-cliquez sur le composant **tJava** pour afficher sa vue **Component**.
- Dans l'onglet des **Basic settings**, saisissez votre code, comme dans la capture d'écran. Ce code permet de vérifier si une chaîne de caractères correspond à une adresse e-mail, en se basant sur l'expression régulière suivante :
  `"^[\\w_.-]+@[\\w_.-]+\\.[\\w]+$"`.

- Appuyez sur **F6** afin d'enregistrer et d'exécuter votre Job.

```
Starting job LibraryLoad at 16:51 18/02/2010.

[statistics] connecting to socket on port 3979
[statistics] connected
ma@cococom : false
[statistics] disconnected
Job LibraryLoad ended at 16:51 18/02/2010. [exit code=0]
```

La **Console** affiche *false*, c'est-à-dire que l'adresse e-mail n'est pas valide. En effet, son format n'est pas correct.

Composants Data Quality

Ce chapitre passe en revue les propriétés des principaux composants présents dans la famille **Data Quality** de la **Palette** de **Talend Open Studio**.

La famille **Data Quality** regroupe les composants vous permettant d'améliorer la qualité de vos données. Ces composants couvrent divers besoin dont la comparaison de colonnes afin d'en extraire des données spécifiques.

tAddCRCRow

Propriétés du tAddCRCRow

Famille de composant	Data Quality	
Fonction	Ce composant calcule une clé de substitution basée sur une ou plusieurs colonne(s) et l'ajoute au schéma défini.	
Objectif	Fournir un ID unique afin d'améliorer la qualité des données traitées.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Implication	Cochez la case correspondant aux colonnes à définir comme clé de substitution et vérifier l'intégrité de leurs données.
Advanced settings	*CRC type*	Sélectionnez la longueur CRC. Plus le CRC est long, moins il y aura de risque d'erreur.
	tStatCatcher Statistics	Cochez cette case pour collecter les métadonnées de process du Job, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Ce composant est une étape intermédiaire. Cela requiert un flux entrant et une sortie.	
Limitation	n/a	

Scénario : Ajouter une clé de substitution à un fichier

Ce scénario décrit un Job ajoutant une clé de substitution à un schéma de fichier délimité.

- Glissez les composants suivants dans l'espace de modélisation: **tFileInputDelimited**, **tAddCRCRow** et **tLogRow**.

- Connectez-les en utilisant un lien de type **Main row**.
- Dans la vue **Component** du **tFileInputDelimited**, renseignez le champ **File Name** en saisissant le chemin d'accès à votre fichier, ainsi que toutes les propriétés en rapport, au cas où elles ne soient pas stockées dans le **Repository**.

- Créez le schéma en cliquant sur le bouton **Edit Schema**, si le schéma n'est pas encore stocké dans le **Repository**. En Java, vérifiez la colonne du type de données, et pour remplir le Date pattern, visitez http://java.sun.com/j2se/1.5.0/docs/api/index.html.
- Dans la vue **Component** du composant **tAddCRCRow**, sélectionnez les cases de la colonne du flux entrant à utiliser pour calculer le CRC.

- Notez qu'une colonne CRC (en lecture seule) a été ajoutée à la fin du schéma.
- Dans les **Advanced Settings**, sélectionnez **CRC32** comme **CRC Type** pour avoir une clé de substitution plus longue.

- Dans la vue du **tLogRow** des **Basic settings**, sélectionnez la case **Print values in cells of a table** pour afficher les données de sortie dans une table de la Console.

- Sauvegardez votre Job et appuyer sur **F6** pour l'exécuter.

```
Starting job addcrc at 11:51 06/07/2007.
.---------+----------+----------+------+-----------+----------.
|                          tLogRow_1                           |
|=--------+----------+----------+------+-----------+----------=|
|ID_Owners|Reg_Car   |Make      |Color |ID_Reseller|CRC       |
|=--------+----------+----------+------+-----------+----------=|
|1        |1301 DO 05|Citroen   |gold  |38         |27510715125|
|2        |2300 ZP 14|Citroen   |blue  |16         |33211434545|
|3        |4122 JI 74|Renault   |yellow|36         |11525215315|
|4        |3395 QP 05|Citroen   |yellow|51         |14306204562|
|5        |0029 OF 61|Toyota    |red   |37         |10711350076|
|6        |4287 YU 44|Citroen   |blue  |43         |25561510712|
|7        |7119 CQ 97|Honda     |yellow|65         |10136571035|
|8        |3764 PA 47|Renault   |orange|30         |31723253034|
|9        |9939 CJ 88|Mercedes  |red   |41         |27451544441|
|10       |7476 RV 09|Citroen   |grey  |34         |27775721061|
|11       |5287 BP 14|Toyota    |green |27         |2716661270 |
|12       |0750 OG 65|Toyota    |green |8          |23636130023|
|13       |7577 ZQ 59|Volkswagen|purple|55         |37277337005|
```

Une colonne CRC supplémentaire a été ajoutée au schéma, calculée d'après toutes les colonnes précédemment sélectionnées (dans ce cas toutes les colonnes du schéma).

tExtractRegexFields

Le **tExtractRegexFields** appartient à deux familles : Data Quality et Processing. Pour plus d'informations concernant le composant **tExtractRegexField**, consultez *tExtractRegexFields, page 1283*.

tFuzzyMatch

Propriétés du tFuzzyMatch

Famille de composant	Data Quality	
Fonction	Le composant **tFuzzyMatch** compare une colonne du flux principal avec une colonne du flux de référence et extrait les données du flux principal en affichant la distance.	
Objectif	Le **tFuzzyMatch** permet d'assurer la qualité des données de n'importe quelle source de données en la comparant avec une source de données de référence.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Deux colonnes en lecture seule sont ajoutées automatiquement au schéma de sortie : Value et Match.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Matching type	Sélectionnez l'algorithme de correspondance approprié parmi les suivants : **Levenshtein** : Basé sur la théorie de la distance d'édition. Il calcule le nombre d'insertion, de suppression ou de substitution nécessaire pour correspondre à la référence. **Metaphone** : Basé sur la phonétique. Tout d'abord, il charge les sonorités de toutes les entrées du flux de référence (Lookup) et les compare avec celles des entrées du flux principal (Main). **Double Metaphone** : une nouvelle version de l'algorithme phonétique Metaphone, qui produit des résultats plus précis que l'algorithme original. Il peut retourner à la fois un code primaire et secondaire pour une chaîne de caractères. Pour éviter toute ambiguïté que le Metaphone n'aurait pas pu détecter, utilisez cette option.

	Min Distance	(Levenshtein uniquement) Définissez le nombre minimum de modifications autorisées pour correspondre à la référence. Si vous définissez 0, seules les correspondances exactes sont extraites.
	Max Distance	(Levenshtein uniquement) Définissez le nombre maximum de modificiations pouvant correspondre à la référence.
	Matching Column	Sélectionnez la colonne du flux principal qui doit être comparée avec la colonne clé du flux de référence (Lookup).
	Unique Matching	Cochez cette case pour obtenir la meilleure correspondance possible, si plusieurs correspondances sont disponibles.
	Matching item separator	Si plusieurs correspondances sont disponibles, elles seront toutes affichées sauf si la case **Unique match** est cochée. Définissez le délimiteur entre chaque correspondance.
Utilisation	Ce composant n'est pas un composant de début (fond vert) et il nécessite deux composants d'entrée et un composant de sortie.	
Limitation/Conditions requises	Utilisateurs Perl : Assurez-vous que le package correspondant est installé sur votre ordinateur. Consultez la vue Module pour connaître les modules à installer.	

Scénario 1 : Distance de Levenshtein de 0 pour les prénoms

Ce scénario décrit un Job constitué de quatre composants dont le but est de vérifier la distance d'édition comprise entre les colonnes *First Name* (prénom) d'un fichier d'entrée et de comparer les données avec celles du fichier de référence. Le résultat de cette vérification de la distance de Levenshtein et le contenu du flux principal sont affichés dans une table.

- Cliquez et déposez les composants suivants de la Palette dans l'espace de modélisation : **tFileInputDelimited** (x2), **tFuzzyMatch** et **tFileOutputDelimited**.
- Paramétrez les propriétés du composant **tFileInputDelimited** à partir de l'onglet **Basic settings**. Parcourez votre répertoire et sélectionnez le fichier d'entrée à analyser. Mais surtout définissez le schéma à utiliser par le flux à vérifier.
- Dans le schéma, définissez le **Type** de données en version Java, surtout si vous êtes en mode **Built-in**.

- Connectez ce composant d'entrée au **tFuzzyMatch** à l'aide d'une connexion de type **Main Row**.
- Définissez le deuxième composant **tFileInputDelimited** de la même manière.

Assurez-vous que la colonne de référence a bien été définie comme colonne clé dans le schéma du flux de référence (lookup).

- Puis connectez le deuxième composant d'entrée au **tFuzzyMatch** à l'aide d'une connexion main row (qui apparaît comme une connexion **Lookup** row dans l'espace de modélisation).
- Définissez les propriétés du **tFuzzyMatch** dans l'onglet **Basic settings**.
- Le **Schéma** doit correspondre à celui du flux d'entrée principal (**Main**) afin que le flux principal soit comparé au flux de référence.

- Notez que deux colonnes : **Value** et **Matching** sont ajoutées au schéma de sortie. Elles correspondent à des informations standard de correspondance et sont en lecture seule.
- Sélectionnez la méthode à utiliser pour la vérification des données entrantes. Dans ce scénario, la correspondance à sélectionner dans le champ **Matching type** est de type *Levenshtein*.
- Ensuite définissez la distance. Dans cette méthode, la distance est le nombre de caractères modifiés (insertion, suppression ou substitution) pour que l'entrée principale corresponde exactement à l'entrée de référence.

- Dans ce scénario, l'objectif est de faire en sorte que la distance soit de 0 en minimum et en maximum dans les champs **Min. distance** et **Max. distance**. Ce qui signifie que seule une correspondance exacte sera redirigée en flux de sortie.
- Décochez la case **Case sensitive** pour ne pas prendre en compte la casse.
- Et dans le champ **Matching column**, sélectionnez la colonne du flux principal qui doit être vérifiée. Dans cet exemple, le prénom (*First name*).
- Ne cochez pas la case **Unique matching** et ne modifiez pas le séparateur.
- Connectez le **tFuzzyMatch** à un composant de sortie standard type **tLogRow**. Ne définissez aucun autre paramètre que le délimiteur pour ce scénario.
- Enregistrez le Job et appuyez sur la touche **F6** pour exécuter le Job.

```
audra (1)||
audra (2)||
audrea||
audrey||
august (1)||
august (2)||
augusta||
auguste|0|auguste
augustijn||
augustin|0|augustin
augustine||
augusto||
augusts||
augustus||
aukusti||
```

Comme la distance d'édition définie est de 0 (min. et max.), la sortie correspond à une jointure entre le flux principal et le flux de référence (lookup), ainsi seules les correspondances totales avec une valeur égale à 0 sont affichées.

Pour un exemple plus évident avec une distance minimum de 1 et une distance maximum de 2, consultez *Scénario 2 : Distance de Levenshtein de 1 ou 2 pour les prénoms, page 151*.

Scénario 2 : Distance de Levenshtein de 1 ou 2 pour les prénoms

Ce scénario est basé sur le scénario 1 décrit ci-dessus. Seuls les paramètres distance min. et max. du composant **tFuzzyMatch** sont modifiés, ce qui modifiera la sortie affichée.

- Dans la vue **Component** du **tFuzzyMatch**, modifiez la distance min. de 0 à 1. Ceci exclut directement les correspondances exactes (qui ont une distance de 0).
- Maintenant mettez une distance max. de 2 puisque la distance max. ne peut être inférieure à la distance min. La sortie correspond à toutes les données mises en correspondance contenant une différence de 2 caractères au plus.

- Aucune autre modification de paramètres n'est nécessaire.
- Assurez-vous que le séparateur de champs (**Matching item separator**) est bien défini, vu que plusieurs entrées de référence peuvent correspondre à l'entrée du flux principal.
- Enregitrez votre nouveau Job et appuyez sur **F6** pour l'exécuter.

```
audrea|2|aude
audrey|2|aude
august (1)||
august (2)||
augusta|1|auguste
auguste|2|augustin
augustijn|1|augustin
augustin|2|auguste
augustine|1|augustin
augusto|1|auguste
augusts|1|auguste
augustus|2|auguste,augustin
aukusti|2|auguste,augustin
aulay||
aulus|2|jules
```

Etant donné que la distance d'édition a été définie à 2, certaines entrées du flux principal correspondent à plusieurs entrées de référence (lookup).

Vous pouvez utiliser une autre méthode, la Métaphone, pour évaluer la distance entre le flux principal et le flux de référence.

Scénario 3 : Distance métaphonique pour les prénoms

Ce scénario est basé sur le scénario 1 décrit plus haut.

- Modifiez le champ **Matching type** avec l'option **Metaphone**. Aucune distance minimum et aucune distance maximum ne sont à définir puisque cette méthode est basée sur les différences de phonétique avec la référence.

- Enregistrez le Job et appuyez sur **F6**. La valeur phonétique est affichée avec les correspondances possibles.

```
audrey||
august (1)|AKST|auguste
august (2)|AKST|auguste
augusta|AKST|auguste
auguste|AKST|auguste
augustijn||
augustin|AKSTN|augustin
augustine|AKSTN|augustin
augusto|AKST|auguste
augusts||
augustus||
aukusti|AKST|auguste
aulay||
aulus||
aune||
```

tIntervalMatch

Propriétés du tIntervalMatch

Famille de composant	Data Quality	
Fonction	Le composant **tIntervalMatch** permet d'établir une jointure entre un flux principal et un flux de référence (en Java) ou un fichier de référence (en Perl). Il classe une valeur spécifiée du flux principal dans la fourchette de valeurs correspondante définie dans le flux de référence et retourne les informations correspondantes.	
Objectif	Il aide à retourner une valeur à l'aide d'une jointure.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (**Built-in**) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous effectuez des modifications, le schéma passera automatiquement en Built-in.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Java only	*Search column*	Sélectionnez la colonne du flux principal contenant les valeurs qui doivent être mises en correspondance avec l'intervalle de valeurs.
	Column (LOOKUP)	Sélectionnez la colonne du flux lookup contenant les valeurs qui doivent être retournées quand la jointure est ok.
	Lookup Column min/ bounds strictly (min)	Sélectionnez la colonne contenant la valeur minimale de l'intervalle. Cliquez sur la case si la limite est stricte.
	Lookup Column max/ bounds strictly (max)	Sélectionnez la colonne contenant la valeur maximale de l'intervalle. Cliquez sur la case si la limite est stricte
Utilisation	Ce composant manipule un flux de données, il requiert donc un Input et un Output, et nécessite la définition d'une étape intermédiaire.	
Limitation	n/a	

Les propriétés en Perl sont trop différentes de celles en Java, elles sont donc décrites dans le tableau ci-dessous..

PERL basic settings	

Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (**Built-in**) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous effectuez des modifications, le schéma passera automatiquement en Built-in.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Perl only	*File Name*	Saisissez le nom du fichier contenant l'intervalle de données. Cela fonctionne comme un flux lookup.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Row separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Lookup index Column	Position de la colonne **min** dans le fichier lookup: 0 pour la première colonne, 1 pour la seconde colonne, etc. Vérifiez que les colonnes min et max de l'intervalle soient adjacentes.
	Search column	Sélectionnez la colonne du flux principal contenant les valeurs qui doivent être mises en correspondance avec l'intervalle de valeurs.
Usage	Ce composant manipule un flux de données, il requiert donc un Input et un Output, et nécessite la définition d'une étape intermédiaire. .	
Limitation	Pour le moment, la version Perl du **tIntervalMatch** n'accepte pas un vrai flux lookup (mais seulement un fichier de référence dans les paramètres actuels du composant).	

Scénario : Identifier le pays à partir de l'IP (Perl et Java)

Le scénario suivant décrit un Job conçu parallèlement dans les deux langages, Perl et Java. Dans ce Job, un flux principal entrant transmet un schéma à deux colonnes: *Documents* et *IP*, contenant des données factices. Un second fichier est utilisé comme flux lookup en Java, comme fichier de référence en Perl, et contient une liste classée d'intervalles d'IP ainsi que le pays qui leur correspond. Ce Job a pour but de retrouver les pays de chaque document à partir de la valeur de leur IP, en d'autres termes, de créer une jointure entre le flux principal et le flux de référence.

En **Perl**, le Job requiert un **tFileInputDelimited**, un **tIntervalMatch** et un **tLogRow**.

En **Java**, le Job requiert un **tFileInputDelimited** supplémentaire, un **tIntervalMatch** et un **tLogRow**.

- Glissez le composant dans l'espace de modélisation.
- Configurez les paramètres de base dans la vue **Component** du **tFileInputDelimited**.

- Le schéma se compose de deux colonnes, respectivement *Document* et *IP*.
- (En Java uniquement) Configurez la colonne **Type** en **String** pour la colonne *Document* et en **Integer** pour la colonne *IP*.
- (En Java uniquement) Configurez maintenant les propriétés du deuxième **tFileInputDelimited**.

- (En Java uniquement) N'oubliez pas de définir le **Type** des données.
- (En Java et en Perl) Propagez le schéma du flux principal entrant au composant **tIntervalMatch**.

- (En Java et en Perl) Notez que le schéma Output du composant **tIntervalMatch** est en lecture seule et est composé du schéma de l'Input ainsi que d'une colonne supplémentaire lookup qui redirigera en sortie les données lookup demandées.
- Configurez les autres propriétés du composant **tIntervalMatch**.
- (En Perl uniquement) Le fichier lookup est défini directement dans les propriétés du **tIntervalMatch**.

- (En Perl uniquement) Dans le champ **File Name**, renseignez le chemin d'accès au fichier lookup. Renseignez le **Row** et le **Field separator** du fichier lookup.
- (En Perl uniquement) Dans le champ **Lookup column index**, paramétrez la limite inférieure de l'intervalle des données. Cela correspond à la position de la colonne contenant la valeur minimum de l'intervalle (0 pour la première colonne).
- (En Perl uniquement) Il n'y a pas besoin de configurer les valeurs lookup à retourner puisque toutes les valeurs du fichier lookup seront transmises en sortie.

- (En Java uniquement) Configurez les autres propriétés du **tIntervalMatch**, notamment les colonnes **min** et **max** correspondant aux limites de l'intervalle.

- (En Java uniquement) Dans le champ **Column Lookup**, sélectionnez la colonne dans laquelle les valeurs devront être retournées.
- (En Java et en Perl) Dans le champ **Search column**, sélectionnez la colonne du flux principal contenant les valeurs qui devront correspondre aux valeurs de l'intervalle.
- (En Java et en Perl) Le composant **tLogRow** ne requiert pas de configuration particulière pour ce Job.

Les Jobs en Perl et en Java écrivent le même résultat avec des légères différences dans la présentation.

```
Starting job IntervalN at 17:03 01/02/2008.
document1|1200|1001;2000;DEUTSCHLAND
document2|4500
document3|500|0;1000;FRANCE
document1|2201|2001;3000;ITALIA
Job IntervalN ended at 17:03 01/02/2008. [exit code=0
```

```
Starting job intervalMatch at 17:05 01/02/2008.
document1|1200|DEUTSCHLAND
document2|4500|
document3|500|FRANCE
document1|2201|ITALIA
Job intervalMatch ended at 17:05 01/02/2008. [exit c
```

Le résultat en Perl inclut l'intervalle de valeurs alors que la sortie Java inclut seulement les valeurs demandées retournées (pays).

tParseAddress

Propriétés du tParseAddress

Famille de composant	Data Quality	
Fonction	Ce composant analyse les adresses dans une colonne définie du schéma et les découpe par types de champs.	
Objectif	Découper les données en différentes colonnes pour trier les différentes parties afin d'améliorer la qualité des données traitées.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Column to parse	Sélectionnez la colonne dont vous souhaitez analyser et décomposer les données.
	Country	Sélectionnez le pays d'où proviennent les clients.
Advanced settings	*Correct case*	Cochez cette case pour que le composant corrige la casse, c'est-à-dire qu'il mette, au besoin, une majuscule en début de nom, et le reste en minuscules.
	Auto clean	Cochez cette case pour commencer par supprimer les caractères non alphanumériques avant d'essayer de décomposer les adresses.
	Abbreviate subcountry	Cochez cette case pour abréger automatiquement le nom des régions ou provinces. Si le nom est déjà sous sa forme abrégé, il ne sera pas modifié.
	Allow only abbreviated subcountry	Cochez cette case pour n'autoriser que les noms de régions ou provinces abrégés. Le traitement des données sera plus rapide et vérifiera que l'adresse correspond bien aux standards postaux.
	tStatCatcher Statistics	Cochez cette case pour collecter les métadonnées de process du Job, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Ce composant est une étape intermédiaire. Cela requiert un flux entrant et une sortie.	
Limitation	n/a	

Scénario associé

Aucun scénario n’est disponible pour ce composant.

tParseName

Propriétés du tParseName

Famille de composant	Data Quality	
Fonction	Ce composant récupère des noms sous différentes formes et en extrait chaque élément en fonction de son type.	
Objectif	Analyser un fichier contenant des noms et en extraire les éléments selon leur type, afin d'améliorer la qualité des données traitées, en rejetant les éléments erronés.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Column to parse	Sélectionnez la colonne dont vous souhaitez extraire les différents éléments.
Advanced settings	*Ignore joint names. Mr John Smith and Ms Mary Jones -> Mr John Smith*	Cochez cette case pour ignorer les données contenant des noms joints. Cocher cette option permet un gain de rapidité. Décochez-la pour que ces données soient retournées et découpées par type.
	Allow reversed. Smith, Mr AB -> Mr AB Smith	Cochez cette case pour que les noms entrés en ordre inversé soient traités. Le composant remet le nom dans l'ordre, et effectue l'analyse normale. Notez que si le nom peut être analysé, son ordre originel n'est pas enregistré comme propriété.
	Extend titles	Cochez cette case afin de pouvoir utiliser tous les titres possibles. Si vous la décochez, vous ne pourrez utiliser que les titres "standards".
	Correct case	Cochez cette case pour que le composant corrige la casse, c'est-à-dire qu'il mette, au besoin, une majuscule en début de nom, et le reste en minuscules.
	Auto clean	Cochez cette case pour commencer par supprimer les caractères non alphanumériques avant d'essayer de décomposer les adresses.

	tStatCatcher Statistics	Cochez cette case pour collecter les métadonnées de process du Job, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Ce composant est une étape intermédiaire. Cela requiert un flux entrant et une sortie.	
Limitation	n/a	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tReplaceList

Propriétés du tReplaceList

Famille de composant	Data Quality	
Fonction	Le **tReplaceList** effectue un rechercher/remplacer dans les colonnes d'entrée spécifiées en fonction des données de référence (lookup).	
Objectif	Ce composant permet de nettoyer tous les fichiers avant un traitement futur.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Deux colonnes en lecture seule, **Value** et **Match** sont automatiquement ajoutés au schéma de sortie. *Le type de données défini dans les schémas doit être cohérent, c'est-à-dire qu'un entier ne peut être remplacé que par un entier en utilisant un entier comme champ de référence. Une valeur d'un type ne peut être remplacée par une valeur d'un autre type.*
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Lookup search column	Saisissez le numéro de la colonne sur laquelle effectuer la recherche dans le schéma de lookup. **0** : la première colonne lue **1** : la deuxième colonne lue **n** : numéro de la colonne dans le schéma lu.
	Afin de s'assurer que les valeurs cherchées sont uniques, assurez-vous que la colonne soit définie comme clé dans votre schéma de lookup.	
	Lookup replacement column	Saisissez le numéro de la colonne dans laquelle les valeurs de remplacement sont stockées. **0** : la première colonne lue **1** : la deuxième colonne lue **n** : numéro de la colonne dans le schéma lu
	Column options	Sélectionnez les colonnes du flux principal dans lequel le remplacement est effectué.

Utilisation	Ce composant est un composant intermédiaire, il nécessite un composant d'entrée et un composant de sortie.

Scénario : Remplacement à partir d'un fichier de référence

Le Job suivant effectue un rechercher/remplacer sur une liste d'Etats avec leur code correspondant. Les codes correspondant sont extraits d'un fichier de référence placé en flux Lookup dans le Job. Le flux principal est dupliqué et les deux sorties sont affichées dans la console, pour constater l'état du flux principal avant et après remplacement.

- Cliquez et déposez les composants suivants de la **Palette** : **tMysqlInput**, **tFileInputDelimited**, **tReplicate**, **tReplaceList** et **tLogRow** (x2). Notez que si vos schémas d'entrée sont déjà centralisés sur le Repository, vous pouvez, d'un simple cliquer-déposer, les faire passer du noeud correspondant du répertoire Metadata du Repository vers l'espace de modélisation. Cela permettra de récupérer automatiquement les propriétés des composants d'entrée. Pour plus d'informations, consultez *Déposer des composants à partir du répertoire Metadata* du **Guide utilisateur** de **Talend Open Studio**.
- Connectez les composants à l'aide de liens **Main Row** via un clic-droit sur chaque composant. Notez que le lien de type main row provenant du flux de référence (**tFileInputDelimited**) se nomme lookup row.
- Sélectionnez le composant **tMysqlInput** et définissez les paramètres du flux d'entrée.

- Le schéma d'entrée est composé de deux colonnes : *Names*, *States*. La colonne States regroupe le nom des Etats des Etats-Unis qui seront remplacés par leur code respectif.
- Dans le champ **Query**, assurez-vous que la colonne *State* est incluse dans la déclaration `Select`. Dans cet exemple, toutes les colonnes sont sélectionnées.
- Vérifiez les paramètres du composant **tReplicate**. Le schéma est simplement divisé en deux flux identiques, mais aucun changement n'a été effectué sur le schéma.
- Puis double-cliquez sur le composant **tFileInputDelimited**, pour paramétrer le fichier de référence.

- Le fichier comporte deux colonnes : *Postal* et *State*. *Postal* correspond au code postal de l'Etat de la colonne *State* correspondante.
- Les champs sont délimités par des point-virgules et les lignes par des retours chariot.
- Modifiez le schéma du flux lookup.

- Assurez-vous que la colonne lookup sur laquelle est effectuée la recherche (dans cet exemple : *State*) est définie comme clé, pour s'assurer que les données cherchées sont uniques.
- Sélectionnez le composant **tReplaceList** et paramétrez l'opération à effectuer.
- Le schéma est récupéré à partir du composant précédent dans le flux principal.

- Dans le champ **Lookup search index**, renseignez la position de la colonne sur laquelle est effectuée la recherche. Dans cet exemple, *State* est la deuxième colonne du fichier d'entrée lookup. Ainsi, saisissez **1** dans ce champ.
- Dans le champ **Lookup replacement index**, renseignez le numéro de la colonne contenant les valeurs de remplacement, dans cet exemple : *Postal* pour le code postal de l'Etat.
- Dans le tableau **Column options**, sélectionnez la colonne *States* puisque dans cet exemple le nom des Etats doit être remplacé par leur code correspondant.
- Pour les deux composants **tLogRow**, cochez la case **Print values in table cells** pour que les sorties soient plus claires.
- Enregistrez le Job et exécutez-le en appuyant sur **F6**.

```
Starting job ReplaceList at 15:06 22/10/2007.
.-----------------------------------------.
|                tLogRow_3                |
+----------------------+-----------------+
| Name                 | States          |
+----------------------+-----------------+
| William Grant        | Iowa            |
| William Hoover       | New York        |
| Grover Lincoln       | North Dakota    |
| Lyndon Jefferson     | Ohio            |
| Gerald Hayes         | Washington      |
| Benjamin Grant       | Maine           |
| George Pierce        | Connecticut     |
| Jimmy Reagan         | Alaska          |
| Martin Hayes         | Washington      |
| Franklin Jefferson   | Iowa            |
| Andrew Nixon         | New Hampshire   |

.-------------------------------.
|            tLogRow_2          |
+----------------------+--------+
| Name                 | States |
+----------------------+--------+
| William Grant        | IA     |
| William Hoover       | NY     |
| Grover Lincoln       | ND     |
| Lyndon Jefferson     | OH     |
| Gerald Hayes         | WA     |
| Benjamin Grant       | ME     |
| George Pierce        | CT     |
| Jimmy Reagan         | AK     |
| Martin Hayes         | WA     |
| Franklin Jefferson   | IA     |
| Andrew Nixon         | NH     |
'

Job ReplaceList ended at 15:06 22/10/2007. [exit code=0]
```

Le premier flux de sortie affiche la colonne States avec les noms des Etats puisqu'il provient du flux d'entrée principal.

Le deuxième flux affiche la colonne States après que le nom des Etats aient été remplacés par leur codes postaux respectifs.

tSchemaComplianceCheck

Propriétés du tSchemaComplianceCheck

Famille de composant	Data Quality	
Fonction	Ce composant valide toutes les lignes d'entrée en fonction du schéma de référence ou vérifie le type, la nullabilité, la longueur de la ligne en fonction des valeurs de référence. Cette validation peut être appliquée à l'ensemble des lignes ou une partie seulement.	
Objectif	Le **tSchemaComplianceCheck** permet d'assurer la qualité des données de n'importe quelle source de données en les comparant aux données de référence.	
Basic settings	*Base Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le repository. Définissez la structure et la nature des données à traiter.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** **Talend Open Studio**.
Java uniquement	*Use another schema for compliance check*	Définissez un schéma de référence en donnant aux données les propriétés qu'elles sont censées avoir. Ainsi, les données non conformes seront rejetées. Ces propriétés peuvent correspondre au type de données, aux valeurs nulles, et/ou à la longueur des données.
Perl uniquement	*Date language/ Date format*	Pour la validation du format de date du type 25 Dec 2007, utilisez le champ Date Language et pour distinguer l'ordre d'affichage du mois et du jour, utilisez le champ Date format.
	Check all columns from schema	Cochez cette case, si vous souhaitez effectuer la vérification sur toutes les colonnes du schéma de base.
	Type	En **Perl**, cochez la case de la colonne dont vous souhaitez vérifier les données en fonction du schéma de base défini. En **Java**, sélectionnez les propriétés que les données doivent avoir. Cette validation est obligatoire pour toutes les colonnes.
	Null (empty or zero)	Cochez la case pour vérifier la nullabilité d'une colonne en fonction de la définition du schéma de base.
	Max length	Cochez la case pour vérifier la longueur des données en fonction de la longueur définie dans le schéma de base.

Utilisation	Ce composant est un composant intermédiaire permettant d'exclure les données non conformes du flux principal. Ce composant ne peut être utilisé comme composant de début puisqu'il requiert un flux d'entrée. Il nécessite aussi au moins un composant de sortie afin de regrouper les flux validés et éventuellement un deuxième pour les données rejetées via un lien Rejects. Pour plus d'informations, consultez *Types de connexions* du **Guide utilisateur** de **Talend Open Studio**.

Scénario : Valider les dates en fonction d'un schéma (java)

Ce scénario très basique permet de vérifier le type, la nullabilité et la longueur des données du flux d'entrée en fonction du schéma de référence défini. Le flux d'entrée provient d'un fichier csv simple contenant des données dont les propriétés ont un mauvais type, un mauvais id, dont les données sont supérieures à la longueur maximum ou les colonnes n'acceptant pas de valeurs nulles en contiennent.

```
1;label 1 with max length 30;another label with max length 40;2007-10-01;not null;nullable;
2;label 2 with max length 30;another label with max length 40 and with length 51;2007-06-31;not null;
3;label 3 with max length exceed 30;another label with max length 40 and with length 51;2007-10-01;;;
4;label 4 with max length 30;another label with max length 40;2007-12-13;not null;;
5;label 5 with max length 30;another label with max length 40;2007-13-12;not null;;
6;label 6 with max length 30;another label with max length 40;13/12/2007;not null;;
7;label 7 with max length 30;another label with max length 40;12/13/2007;not null;;
8.6;label with max length 30;another label with max length 40;2007-10-01;not null;
9;label 9 with max length 30;another label with max length 40;2007-10-01;;;
10;label 10 with max length exceed 30;another label with max length 40 and with length
51;2007-10-01;not null;nullable but not null;
```

La sortie est dupliquée : les données valides sont regroupées dans un même fichier délimité, alors que les données rejetées sont affichées dans la console.

- Cliquez et déposez les composants suivants : **tFileInputDelimited**, **tSchemaComplianceCheck**, **tFileOutputDelimited**, **tLogRow**.
- Cliquez-droit sur le **tFileInputDelimited** pour le connecter au composant **tSchemaComplianceCheck** via une connexion **Row Main**.
- Cliquez-droit sur le composant **tSchemaComplianceCheck** et sélectionnez **Row > Main** pour le connecter au composant **tFileOutputDelimited**. Ce flux de sortie ne contiendra que les données valides.

- Cliquez-droit une deuxième fois sur le composant **tSchemaComplianceCheck** et cette fois sélectionnez **Row > Rejects** pour le connecter au composant **tLogRow**. Cette deuxième sortie regroupera toutes les données non conformes.
- Sélectionnez le lien **Rejects** que vous venez de connecter, vous remarquerez que le schéma transmis au **tLogRow** contient deux nouvelles colonnes : *ErrorCode* et *ErrorMessage*. Ces deux colonnes sont en lecture seule et fournissent des informations sur les données rejetées, afin de faciliter le traitement et la correction de erreurs, si nécessaire.
- Maintenant définissez les propriétés de chaque composant.

- Dans la vue **Component** du composant **tFileInputDelimited**, laissez les paramètres par défaut du fichier délimité et renseignez le chemin d'accès au fichier d'entrée dans le champ **File Name**. Dans cet exemple : *in.csv*.
- Cliquez sur **Edit Schema** pour décrire la structure de données de ce fichier d'entrée. Le schéma est composé de six colonnes : *id*, *Col2Label*, *Col3Label*, *Date*, *Val* et *Str_Nullable*.

- Laissez le champ **Type** aussi permissif que possible (particulièrement en Java). Vous définirez le type de données réel dans la vue **Component** du composant **tSchemaComplianceCheck**.
- La colonne *Str_nullable* est la seule colonne contenant des données de valeur nulle. Ainsi, cochez la case adéquate. Les colonnes *Col2Label* et *Col3Label* ont respectivement une longueur de *30* et *40* caractères.
- Puis double-cliquez sur le composant **tSchemaComplianceCheck** pour configurer les paramètres de validation.

- Le champ **Base schema** devrait être renseigné par les données propagées à partir du composant d'entrée. Si cela n'est pas le cas, cliquez sur le bouton **Sync columns**.
- Dans cet exemple, vous utilisez le tableau **Checked columns** pour configurer les paramètres de validation. Mais vous pouvez aussi cocher la case **Use another schema for compliance check** et définir le schéma des données attendues.
- Cochez la case **Check all columns from schema** si vous souhaitez vérifier tous les paramètres de toutes les colonnes (type, valeurs nulles et longueur).
- Dans le tableau **Checked Columns**, définissez les vérifications à effectuer. Vous trouverez ci-dessous les paramètres pour la création du Job dans **Talend Open Studio** avec Java comme langage de génération :
  - Sélectionnez **Int** comme **Type** de données de la colonne *Id*.
  - Cochez la case **Length** afin que les longueurs des colonnes *Col2Label* et *Col3Label* soient vérifiées.
  - Sélectionnez **Date** comme **Type** de données de la colonne *Date*.
  - Cochez la case **null** de la colonne *Val* afin que les valeurs nulles soient vérifiées (car elles ne devraient pas être nulles).
  - Cochez aussi la case null de la colonne *Str_nullable* afin que les valeurs nulles soient vérifiées (car elles peuvent être nulles).
- En Perl, cochez la case **Type** correspondant à la colonne dont le type doit être vérifié (Date). Définissez aussi le format de date dans la liste déroulante **Date format** et la langue dans laquelle la date apparaîtra dans la liste déroulante **Date language**.
- Puis configurez les paramètres des composants de sortie.
- La schéma est automatiquement propagé du composant **tSchemaComplianceCheck** aux composants de sortie.
- Renseignez le chemin d'accès au fichier de sortie dans les propriétés du composant **tFileOutputDelimited**.
- Puis dans la vue **Run**, cochez la case **Statistics** pour afficher les taux de répartition des flux avant de lancer l'exécution.

- Vous remarquerez que trois lignes du flux d'entrée ont été validées et donc envoyées au composant **tFileOutputDelimited**.
- De plus, sept lignes ne sont pas conformes avec le schéma de référence et ont donc été redirigées vers la console.

```
Starting job compliancecheck2 at 11:53 18/12/2007.
[statistics] connecting to socket on port 3919
[statistics] connected
2|label 2 with max length 30|another label with max length 40 and with length
51|2007-06-31|not null||16|Col3Label:exceed max length;Date:Date format not valid
3|label 3 with max length exceed 30|another label with max length 40 and with length
51|2007-10-01|||16|Col2Label:exceed max length;Col3Label:exceed max length
5|label 5 with max length 30|another label with max length 40|2007-13-12|not
null||2|Date:Date format not valid
6|label 6 with max length 30|another label with max length 40|13/12/2007|not
null||2|Date:Date format not valid
7|label 7 with max length 30|another label with max length 40|12/13/2007|not
null||2|Date:Date format not valid
8.6|label with max length 30|another label with max length 40|2007-10-01|not null||2|id:wrong
type
10|label 10 with max length exceed 30|another label with max length 40 and with length
51|2007-10-01|not null|nullable but not null|16|Col2Label:exceed max length;Col3Label:exceed
max length
[statistics] disconnected
Job compliancecheck2 ended at 11:53 18/12/2007. [exit code=0]
```

Vous remarquerez que les colonnes de sortie supplémentaires du lien **Rejects** fournissent les informations **error code** et **error message**, notamment : *2|Date:Date format not valid* or *2|id:wrong type* afin de faciliter l'identification de l'erreur.

tUniqRow

Propriétés du tUniqRow

Famille de composant	Data Quality	
Fonction	Le composant **tUniqRow** compare les entrées et supprime les doublons du flux d'entrée.	
Objectif	Ce composant assure une qualité de données des flux d'entrée et de sortie du Job.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous effectuez des modifications, le schéma passera automatiquement en built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent dans le Job. *Si vous souhaitez supprimer les doublons sur des colonnes en particulier, paramétrez-les dans le schéma.*
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également ; *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Case sensitive	Cochez cette case pour tenir compte des majuscules et des minuscules.
Utilisation	Ce composant est un composant intermédiaire puisqu'il sert à traiter un flux de données. Ainsi, il nécessite un composant d'entrée et un composant de sortie.	
Limitation	n/a	

Scénario : Dédoublonner des données

Ce scénario est basé sur celui du **tSortRow**, le composant **tUniqRow** est ajouté au Job afin de supprimer les doublons dans le flux de sortie. En effet, comme les données en entrée sont créées de manière aléatoire, des doublons sont forcément créés.

- Dans l'onglet **Basic settings** de la vue **Component** du composant **tUniqRow**, cliquez sur **Edit Schema...** pour définir le champ *Names* comme clé et ainsi supprimer les doublons du flux de sortie sur ce critère.
- Cochez la case **Case Sensitive** pour tenir compte de la casse.
- Appuyez sur **F6** pour exécuter le Job. La console affiche les résultats triés et sans doublons.

```
Starting job UnduplicateJob at 11:02 03/01/2007.
5|3|Mickael|93
2|1|Pierrick|92
8|1|Steffie|89
4|3|Fabrice|75
3|4|Bertrand|67
7|1|Matthew|28
Job UnduplicateJob ended at 11:02 03/01/2007. [exit code=0]
```

Composants Databases

Ce chapitre passe en revue les propriétés des principaux composants présents dans la famille **Databases** de la **Palette** de **Talend Open Studio**.

La famille **Databases** regroupe les connecteurs de bases de données les plus populaires. Ces connecteurs couvrent divers besoins : ouverture de connexion, lecture et écriture de tables, commit de transactions, ainsi que des rollbacks pour le traitement des erreurs. Plus de 40 SGBDR sont supportés.

tAccessBulkExec

Propriétés du tAccessBulkExec

Les composants **tAccessOutputBulk** et **tAccessBulkExec** sont généralement utilisés ensemble pour écrire des données dans un fichier délimité puis effectuer différentes actions sur le fichier dans une base de données Access, processus en deux étapes. Ces deux étapes sont regroupées dans le composant **tAccessOutputBulkExec**, détaillé dans une section séparée. L'intérêt d'utiliser un processus en deux étapes réside dans le fait que cela permet de procéder à des transformations avant le chargement des données dans la base de données.

Famille de composant	Databases/Access	
Fonction	Le composant **tAccessOutputBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tAccessBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données Access	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et cliquez sur le composant adéquat à partir de la liste **Component** pour réutiliser les paramètres d'une connexion que vous avez déjà définie. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existante dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexionà travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de bases de données que vous utilisez. Sinon, vous pouvez également désactiver les composants de connexion et utiliser les **Dynamic settings** du composant afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings)**, consultez le **Guide Utilisateur** de votre studio.
	Host	Adresse IP du serveur de base de données.

	Port	Numéro de port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username and *Password*	Informations d'authentification de l'utilisateur de base de données.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create table** : supprimer la table puis en créer une nouvelle. **Create table** : créer une table qui n'existe pas encore. **Create table if not exists** : créer la table si nécessaire. **Clear table** : supprimer le contenu de la table. **Truncate table** : supprimer le contenu de la table mais sans possibilité de Rollback.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit exister pour que l'opération Insert soit authorisée.
	Local filename	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Action on data	Vous pouvez effectuer, sur les données de la table définie, l'opération que vous souhaitez : **Insert** : ajoute de nouvelles entrées à la table.
	Schema et *Edit Schema*	Un schéma est une description de ligne, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *How to set a built-in schema* of **Talend Open Studio User Guide**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur Talend Open Studio**.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Include header	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Usage	Ce composant est généralement utilisé avec le composant **tAccessOutputBulk**. Ensemble, ils offrent un gain de performance important pour l'alimentation d'une base de donnés Access.

Scénarios associés

Pour plus d'informations relatives au fonctionnement du composant **tAccessBulkExec**, consultez les scénarios suivants :

- **tMysqlOutputBulk** *Scénario : Insérer des données transformées dans une base MySQL, page 514*
- **tMysqlOutputBulkExec** *Scénario : Insérer des données dans une base MySQL, page 520*

tAccessInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tAccessInput

Famille de composant	Databases/Access	
Fonction	Le composant **tAccessInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tAccessInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	DB Version	Sélectionnez la version d'Access que vous utilisez
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant (Repository). Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.

		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Query type et Query	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Additional JDBC parameters*	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimer les espaces en début et en fin de champ dans les colonnes sélectionnées.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez les scénarios **tDBInput** et **tMysqlInput** :

- le *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239*.
- le *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241*.
- le *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478*.

Pour plus d'informations sur les contextes dynamiques dans le cadre d'une utilisation de base de données, consultez le composant *tContextLoad, page 1189*.

tAccessOutput

Propriétés du tAccessOutput

Famille de composant	Databases/Access	
Fonction	Le composant **tAccessOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tAccessOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio** .
	DB Version	Sélectionnez la version d'Access que vous utilisez
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC parameters*	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données.

	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tAccessOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	Support null in "SQL WHERE" statement	Cochez cette case pour prendre en compte les valeurs Null d'une table de base de données. Assurez-vous que la case Nullable est bien cochée pour les colonnes du schéma correspondantes.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données Access. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour un scénario associé, consultez :

- Le *Scénario : Afficher la base de données en sortie, page 245* du **tDBOutput**.

- Le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du **tMySQLOutput**.

tAccessOutputBulk

Propriétés du tAccessOutputBulk

Les composants **tAccessOutputBulk** et **tAccessBulkExec** sont généralement utilisés ensemble pour écrire des données dans un fichier délimité puis effectuer différentes actions sur le fichier dans une base de données Access, processus en deux étapes. Ces deux étapes sont regroupées dans le composant **tAccessOutputBulkExec**, détaillé dans une section séparée. L'intérêt d'utiliser un processus en deux étapes réside dans le fait que cela permet de procéder à des transformations avant le chargement des données dans la base de données.

Famille de composant	Databases/Access	
Fonction	Le composant **tAccessOutputBulk** écrit un fichier délimité.	
Objectif	Prépare le fichier à utiliser comme paramètre dans la requête INSERT servant à alimenter une base de données Access.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
	Create directory if not exists	Cette case est cochée par défaut. Cette option permet de créer le dossier contenant le fichier de sortie s'il n'existe pas déjà.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
Advanced settings	*Include header*	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.

	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec le composant **tAccessBulkExec**. Ensemble, ils offrent un gain de performance important pour l'alimentation d'une base de données Access.	

Scénarios associés

Pour un scénario associé au **tAccessOutputBulk**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du composant **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.

tAccessOutputBulkExec

Propriétés du tAccessOutputBulkExec

Les composants **tAccessOutputBulk** et **tAccessBulkExec** sont généralement utilisés ensemble afin d'écrire des données dans un fichier délimité, puis d'effectuer différentes actions sur le fichier dans une base de données Access, processus en deux étapes. Ces deux étapes sont regroupées dans le **tAccessOutputBulkExec**.

Famille de composant	Databases/Access	
Fonction	Le **tAccessOutputBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tAccessOutputBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données Access.	
Basic settings	*Property Type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	DB Version	Sélectionnez la version d'Access que vous utilisez
	DB Name	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'insert soit autorisée.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
	Action on data	Vous pouvez effectuer, sur les données de la table définie, l'opération que vous souhaitez : **Insert** : ajoute de nouvelles entrées à la table.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Create directory if not exists	Cette case est cochée par défaut. Cette option permet de créer le dossier contenant le fichier de sortie s'il n'existe pas déjà.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
Advanced settings	*Additional JDBC parameters*	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données.
	Include header	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est principalement utilisé lorsqu'aucune transformation particulière n'est requise sur les données à charger dans la base de données.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé au **tAccessOutputBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du composant **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.

tAccessRow

Propriétés du tAccessRow

Famille de composant	Databases/Access	
Fonction	**tAccessRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tAccessRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	DB Version	Sélectionnez la version d'Access que vous utilisez.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à traiter.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.

	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249* du **tDBSQLRow**.
- **le** *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du **tMySQLRow**.

tAS400Close

Propriétés du tAS400Close

Famille de composant	Databases/AS400	
Fonction	Le composant **tAS400Close** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données AS400.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tAS400Connection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tAS400Connection** et **tAS400Rollback**. Il est généralement utilisé avec un composant **tAS400Connection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tAS400Close**, consultez *tMysqlConnection, page 471*

tAS400Commit

Propriétés du tAS400Commit

Ce composant est étroitement lié aux composants **tAS400Connection** et **tAS400Rollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/AS400	
Fonction	Le composant **tAS400Commit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tAS400Connection** dans la liste.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tAS400Commit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants AS400 et notamment avec les composants **tAS400Connection** et **tAS400Rollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tAS400Connection** et **tAS400Rollback**. Ils sont généralement utilisés avec un composant **tAS400Connection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tAS400Commit**, consultez *Propriétés du tMysqlConnection, page 471*

tAS400Connection

Propriétés du tAS400Connection

Ce composant est étroitement lié aux composants **tAS400Commit** et **tAS400Rollback**. Ils sont généralement utilisés avec un composant **tAS400Connection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/AS400	
Fonction	Le composant **tAS400Connection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	DB Version	Sélectionnez la version d'AS400 que vous utilisez.
	Host	Adresse IP du serveur de base de données.
	Database	Nom de la base de données.
	Table Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant.. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Auto commit	Cochez cette case pour que, lorsqu'une déclaration est terminée, la méthode commit soit appelée automatiquement sur cette déclaration.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants AS400 et notamment avec les composants **tAS400Commit** et **tAS400Rollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tAS400Commit** et **tAS400Rollback**. Ils sont généralement utilisés avec un composant **tAS400Connection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tAS400Connection**, consultez *tMysqlConnection, page 471*

tAS400Input

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tAS400Input

Famille de composant	Databases/AS400	
Fonction	Le composant **tAS400Input** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tAS400SInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Use an existing connection*	Cochez cette case et sélectionnez le composant **tAS400SInput** adéquat à partir de la liste **Component List** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio
	Property type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.

		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	DB Version	Sélectionnez la version d'AS400 que vous utilisez.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Query type et Query	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.

	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez les scénarios du composant **tDBInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239.*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241.*
- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478.*

Consultez également le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du composant **tContextLoad**.

tAS400LastInsertId

Propriétés du tAS400LastInsertId

Famille de composant	Databases/AS400	
Fonction	**tAS400LastInsertId** affiche les derniers ID ajoutés à une table à partir d'une connexion AS400 spécifiée.	
Objectif	**tMysqlLastInsertId** permet de récupérer les clés primaires des entrées dernièrement ajoutées par un utilisateur à une table AS400.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Component list	Sélectionnez le composant **tAS400Connection** dans la liste s'il y a plus d'une connexion dans votre Job.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est un composant intermédiaire.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Récupérer les ID des dernières entrées ajoutées, page 484* du composant **tMysqlLastInsertId**.

tAS400Output

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tAS400Output

Famille de composant	Databases/DB2	
Fonction	Le composant **tAS400Output** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tAS400Output** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio** .
	DB Version	Sélectionnez la version d'AS400 que vous utilisez.

	Use an existing connection	Cochez cette case et sélectionnez le composant **tAS400SOutput** adéquat à partir de la liste **Component List** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).* La fonction de schéma dynamique peut être utilisée dans différents modes : **Insert**, **Update**, **Insert or Update**, **Update or insert**, **Delete**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.

		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Use commit control*	Cochez cette case pour accéder au champ **Commit every** et définir les opérations de commit. **Commit every** : nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de transaction (cependant pas de rollback) et surtout une meilleure performance des exécutions.
	Additional JDBC parameters	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tAS400Output** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données AS400. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.

Scénarios associés

Pour des scénarios associés, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput**
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMySQLOutput**.
- le *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507* du composant **tMysqlOutput**.

tAS400Rollback

Propriétés du tAS400Rollback

Ce composant est étroitement lié aux composants **tAS400Commit** et **tAS400Connection**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/AS400	
Fonction	Le **tAS400Rollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant de connexion **tAS400Connection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Il faut utiliser ce composant en association avec des composants AS400, notamment avec le **tAS400Connection** et le **tAS400Commit**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au **tAS400Rollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522* du composant **tMySQLRollback**.

tAS400Row

Propriétés du tAS400Row

Famille de composant	Databases/AS400	
Fonction	**tAS400Row** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tAS400Row** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Use an existing connection*	Cochez cette case et sélectionnez le composant **tAS400Connection** adéquat à partir de la liste **Component** pour réutiliser les paramètres d'une connexion que vous avez déjà définie. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Property type	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	DB Version	Sélectionnez la version d'AS400 que vous utilisez.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.

	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à traiter.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le Repository. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC Parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Propagate QUERY's recordset	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.

	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des “?” dans l’instruction SQL définie dans le champ **Query** de l’onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l’instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l’écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d’exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log lors du traitement du Job, aussi bien au niveau du Job qu’au niveau de chaque composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l’auto-incrémentation d’une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tCreateTable

Propriétés du tCreateTable

Ce composant se trouve à la racine de la famille **Databases** de la **Palette** de **Talend Open Studio**. Le composant **tCreateTable** couvre des besoins indirectement liés à l'utilisation des bases de données de tout type.

Famille de composant	Databases	
Fonction	Le **tCreateTable** crée, supprime ou supprime et recrée une table spécifique.	
Objectif	Ce composant spécifique Java permet de créer ou de supprimer n'importe quelle table d'une base de données	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Database Type	Sélectionnez le type de SGBD dans la liste proposée. Les propriétés peuvent différer légèrement en fonction du type de base de données sélectionné. Les champs additionnels sont définis dans ce tableau, à la suite des propriétés générales.
	Table Action	Sélectionnez l'action à effectuer sur la base de données : **Create table** : lorsque vous savez que la table n'existe pas déjà. **Create table when not exists** : lorsque vous ne savez pas si la table existe déjà. **Drop and create table** : lorsque vous savez que la table existe déjà et que vous souhaitez la remplacer.
Mysql	*Temporary Table*	Cochez cette case pour enregistrer la table créée de manière temporaire.

MSSQLServer, MySQL, Oracle, PostgresPlus, Postgresql, Teradata	*Use an existing connection*	Cochez cette case si vous utilisez un composant de connexion, par exemple : **tMysqlConnection** ou **tOracleConnection**, etc. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
Oracle	*Connection Type*	Liste déroulante des pilotes disponibles.
Access	*Access File*	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
Firebird	*Firebird File*	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
Interbase	*Interbase File*	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
SQLite	*SQLite File*	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
Java Db	*Framework Type*	Sélectionnez dans la liste un framework pour votre base de données Java.
HSQLDb	*Running Mode*	Sélectionnez dans la liste le type de serveur correspondant aux paramètres de votre base de données.
HSQLDb	*Use TLS/SSL Sockets*	Cochez cette case pour autoriser, si besoin, le mode sécurisé.

AS400/Oracle	*DB Version*	Sélectionnez dans la liste déroulante la version correspondant à la base de donnée sélectionnée.
Teradata	*Create*	Sélectionnez la table à partir de la liste déroulante. La table à créer peut être : -**SET TABLE** : table ne permettant pas de dupliquer les lignes -**MULTI SET TABLE** : table permettant de dupliquer les lignes.
Tous les types de bases de données, exceptés Access, JavaDb, SQLite et ODBC	*Host*	Adresse IP du serveur de base de données.
Tous les types de bases de données, exceptés Access, Firebird, HSQLDb, SQLite et ODBC	*Database Name*	Nom de la base de données.
JavaDb	*DB Root Path*	Parcourez vos fichiers jusqu'à la racine de votre base de données.
Tous les types de bases de données, exceptés Access, AS400, Firebird, Interbase, JavaDb, SQLite et ODBC	*Port*	Numéro du port d'écoute du serveur de base de données.
HSQLDb	*DB Alias*	Nom de la base de données.
Informix	*DB Server*	Nom du serveur de base de données.
ODBC	*ODBC Name*	Nom de la base de données.
	UserName et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table name	Saisissez le nom de la nouvelle table entre guillemets.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. *Réinitialisez le type de base de données en cliquant sur le bouton correspondant afin de vous assurer que le type de données est correct.*
		Built-in : Le schéma est créé et conservé pour ce composant uniquement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.

Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log lors du traitement du Job, aussi bien au niveau du Job qu'au niveau de chaque composant.
AS400/ MSSQL Server	*Additional JDBC Parameters*	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données.
Utilisation	Ce composant offre la flexibilité des requêtes de base de données et couvre toutes les possibilités des requêtes SQL. Plus de scénarios sont disponibles pour chaque composant d'entrée de type base de données.	

Scénario : Créer une nouvelle table dans une base de données MySQL

Le Job décrit ci-dessous a été conçu pour créer une nouvelle table dans une base de données, composée d'un schéma pris à partir du schéma d'un fichier délimité stocké dans le Repository. Ce Job est composé d'un seul composant.

- Cliquez et déposez un composant **tCreateTable** de la famille **Databases** de la **Palette**.
- Dans la liste **Database type** de l'onglet **Basic settings** de la vue **Component**, sélectionnez *MySQL*.

- Dans la liste déroulante **Table Action**, sélectionnez **Create table**.
- Ne cochez la case **Use Existing Connection** uniquement si vous utilisez un composant avec une connexion dédiée, consultez *tMysqlConnection, page 471* Vous n'utiliserez pas cette option pour cet exemple.
- Dans la liste déroulante **Property type**, sélectionnez **Repository**, ainsi tous les champs de connexion à la base de données sont renseignés automatiquement. Si vous n'avez pas configuré de connexion aux bases de données dans le dossier **Metadata** du répertoire **DB connection**, renseignez manuellement ces informations de connexion après avoir sélectionné **Built-in** dans la liste déroulante **Property type**.
- Dans le champ **Table Name**, saisissez le nom de la nouvelle table à créer.

- Si vous souhaitez récupérer le schéma d'une métadonnée (sans que celle-ci ne soit une connexion à une base de données), sélectionnez Repository puis la métadonnée souhaitée.
- Dans tous les cas (**Built-in** ou **Repository**), cliquez sur **Edit Schema** pour vérifier le type de données dans la colonne **DB Type**.

- Cliquez sur le bouton **Reset DB Types** si la colonne **DB type** est vide ou affiche des marques de différence (couleur orange). Cela permet de donner à n'importe quel type de données, le type de données adéquat.
- Cliquez sur **OK**.
- Enregistrez votre Job appuyez sur **F6** pour l'exécuter.

La table créée est vide mais elle contient toutes les colonnes définies dans le Schéma.

DB2BulkExec

Propriétés du tDB2BulkExec

Famille de composant	Databases/DB2	
Fonction	Le composant **tDB2BulkExec** exécute des opérations d'Insert sur les données fournies.	
Objectif	Le **tDB2BulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données DB2.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tJDBCConnection** adéquat à partir de la liste **Component List** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Table Schema	Nom du schéma.

	Username et *Password*	Informations d'authentification sur l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Data file	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une clé de recherche sur laquelle baser les opérations Update et Delete. Il est possible de définir les colonnes qui agiront comme clé de recherche à partir du schéma, pour une utilisation de base, ou à partir des options avancées (Advanced settings) pour une utilisation optimisée de ces opérations.*

Advanced settings	*Field terminated by*	Caractère, chaîne ou expression régulière séparant les champs.
	Date Format	Utilisez ce champ pour définir l'ordre des mois et des jours.
	Time Format	Utilisez ce champ pour définir l'ordre des heures, minutes et secondes.
	Timestamp Format	Utilisez ce champ pour définir l'ordre de la date et de l'heure.
	Remove Load Pending	Quand la case est cochée, débloque la table en état "Pending", après un chargement massif.
	Load options	Cliquez sur le bouton **[+]** afin d'ajouter des options de chargement des données. **Parameter** : dans la liste, sélectionnez un paramètre de chargement. **Value** : saisissez une valeur pour le paramètre sélectionné.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes DB2.	

Scénario associé

Pour un scénario associé au composant **tDB2BulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tDB2Close

Propriétés du tDB2Close

Famille de composant	Databases/DB2	
Fonction	Le composant **tDB2Close** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données DB2.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tDB2Connection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tDB2Connection** et **tDB2Rollback**. Il est généralement utilisé avec un composant **tDB2Connection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tDB2Close**, consultez *tMysqlConnection, page 471*

tDB2Commit

Propriétés du tDB2Commit

Ce composant est étroitement lié aux composants **tDB2Connection** et **tDB2Rollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/DB2	
Fonction	Le composant **tDB2Commit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tDB2Connection** dans la liste.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tDB2Commit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants DB2 et notamment **tDB2Connection** et **tDB2Rollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tDB2Connection** et **tDB2Rollback**. Ils sont généralement utilisés avec un composant **tDB2Connection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tDB2Commit**, consultez *tMysqlConnection, page 471*

tDB2Connection

Propriétés du tDB2Connection

Ce composant est étroitement lié aux composants **tDB2Commit** et **tDB2Rollback**. Ils sont généralement utilisés avec un composant **tDB2Connection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/DB2	
Fonction	Le composant **tDB2Connection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier une fois qu'il est validé en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Table Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*Auto commit*	Cochez cette case pour que, lorsqu'une déclaration est terminée, la méthode commit soit appelée automatiquement sur cette déclaration.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant est généralement utilisé avec des composants DB2, notamment les composants **tDB2Commit** et **tDB2Rollback**.
Limitation	n/a

Scénarios associés

Pour un scénario associé au composant **tDB2Connection**, consultez le *Scénario : Insérer des données dans des tables mère/fille, page 471*.

tDB2Input

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tDB2Input

Famille de composant	Databases/DB2	
Fonction	Le composant **tDB2Input** lit une base de données et en extrait des champs à l'aide de requêtes.	
Purpose	Le **tDB2Input** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case lorsque vous utilisez un composant **tDB2Input.** Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur d'annuaire DB2.
	Port	Numéro du port d'écoute du serveur.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.

		Repository : Le schéma existe déjà et il est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table source dans laquelle capturer les changements apportés aux données.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Trim all the String/Char columns*	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données DB2.	

Scénarios associés

Pour un scénario associé, consultez les scénarios du composant **tDBInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239.*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241.*
- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478.*

Consultez également le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du composant **tContextLoad**.

tDB2Output

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tDB2Output

Famille de composant	Databases/DB2	
Fonction	Le composant **tDB2Output** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tDB2Output** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Use an existing connection*	Cochez cette case lorsque vous utilisez un composant **tDB2Output.** Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Property type	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.

		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).* La fonction de schéma dynamique peut être utilisée dans différents modes : **Insert**, **Update**, **Insert or Update**, **Update or insert**, **Delete**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.

		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de transaction (cependant pas de rollback) et surtout une meilleure performance des exécutions.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tDB2Output** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Convert columns and table names to uppercase	Cochez cette case pour mettre le nom des colonnes et celui de la table en majuscules.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	Support null in "SQL WHERE" statement	Cochez cette case pour prendre en compte les valeurs Null d'une table de base de données. Assurez-vous que la case Nullable est bien cochée pour les colonnes du schéma correspondantes.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données DB2. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.

Scénarios associés

Pour un scénario associé au composant **tDB2Output**, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du **tMySQLOutput**.
- le *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507*.

tDB2Rollback

Propriétés du tDB2Rollback

Ce composant est étroitement lié aux composants **tDB2Commit** et **tDB2Connection**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/DB2	
Fonction	Le **tDB2Rollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant de connexion **tDB2Connection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Il faut utiliser ce composant en association avec des composants DB2, notamment avec le **tDB2Connection** et le **tDB2Commit**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au **tDB2Rollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522* du composant **tMySQLRollback**.

tDB2Row

Propriétés du tDB2Row

Famille de composant	Databases/DB2	
Fonction	**tDB2Row** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tDB2Row** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Use an existing connection*	Cochez cette case et sélectionnez le composant **tDB2Connection** adéquat à partir de la liste **Component** pour réutiliser les paramètres d'une connexion que vous avez déjà définie. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Property type	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.

	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à traiter.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le Repository. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.

	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des “?” dans l’instruction SQL définie dans le champ **Query** de l’onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l’instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé au composant **tDB2Row**, consultez :

- le *Scénario : Réinitialiser l’auto-incrémentation d’une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tDB2SCD

Le composant **tDB2SCD** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tDB2SCD, page 2*.

tDB2SCDELT

Le composant **tDB2SCDELT** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tDB2SCDELT, page 4*.

tDB2SP

Propriétés du tDB2SP

Famille de composant	Databases/DB2	
Fonction	Le composant **tDB2SP** appelle une procédure stockée de base de données.	
Objectif	Le **tDB2SP** permet de centraliser des requêtes multiples ou complexes dans une base de données et de les appeler plus facilement.	
Basic settings	*Use an existing connection*	Cochez cette case et sélectionnez le composant **tDB2Connection** adéquat à partir de la liste **Component** pour réutiliser les paramètres d'une connexion que vous avez déjà définie. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Property type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier des propriétés du composant. Les champs suivants sont alors pré-remplis.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification sur l'utilisateur de base de données.

	Schema et *Edit Schema*	Dans une procédure stockée, le schéma est un paramètre d'entrée. Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le **Repository**. Ainsi, il est réutilisable. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	SP Name	Saisissez le nom exact de la Procédure Stockée (SP).
	Is Function / Return result in	Cochez cette case, si une seule valeur doit être retournée. Sélectionnez dans la liste la colonne du schéma sur laquelle est basée la valeur à obtenir.
	Parameters	Cliquez sur le bouton (+) et sélectionnez dans le champ **Schema Columns** les différentes colonnes nécessaires à la procédure. Notez que le schéma de la SP peut contenir plus de colonnes qu'il n'y a de paramètres utilisés dans la procédure. Sélectionnez le **Type** de paramètre : **IN** : paramètre d'entrée (Input) **OUT** : paramètre de sortie (Output)/valeur retournée **IN OUT** : les paramètres d'entrée doivent être retournées sous forme de valeur, même après modifications via la procédure (fonction). **RECORDSET** : les paramètres d'entrée doivent être retournées sous forme d'ensemble de valeurs, au lieu d'une valeur unique. Consultez le composant *tParseRecordSet* si vous voulez analyser un ensemble d'enregistrements d'une table de données ou d'une requête SQL.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est un composant intermédiaire. Il peut être utilisé comme composant de début. Dans ce cas, seuls les paramètres d'entrée sont autorisés.	
Limitation	La syntaxe de la Procédure Stockée doit correspondre à celle de la base de données.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Trouver le libellé State à l'aide d'une procédure stockée, page 537* du composant **tMysqlSP**.

Consultez également le composant *tParseRecordSet, page 635* si vous voulez analyser un ensemble d'enregistrements d'une table de données ou d'une requête SQL.

tDBInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tDBInput

Famille de composant	Databases/DB Generic	
Fonction	**Le composant tDBInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tDBInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row). Préférez toujours un composant Input spécifique au composant générique	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Connection type	Liste déroulante des pilotes disponibles.
	Database	Nom de la base de données
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à traiter.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le Repository. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
Advanced settings	*Trim all the String/Char columns*	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL grâce à une connexion générique ODBC.	

Scénario 1 : Afficher les données sélectionnées à partir d'une table

Le Job suivant, à deux composants, lit les données d'une base via une requête et affiche dans la console (vue **Run**) les données délimitées.

- Cliquez et déposez les composants **tDBInput** et **tLogRow** dans l'éditeur graphique.
- Cliquez droit sur le **tDBInput** et, en gardant le bouton droit enfoncé, glissez jusqu'au **tLogRow** pour créer une connexion de type **Main Row**.
- Sélectionnez à nouveau le **tDBInput** pour que la vue **Component** apparaisse puis renseignez les propriétés du composant :

- Dans ce scénario, le champ **Property Type** est en mode **Built-In**.
- Renseignez les informations de connexion à la base de données dans les champs **Database**, **Username** et **Password**.
- Le champ **Schema** est en mode **Built-In**, ce qui implique que le schéma ne sera utilisable que pour ce Job et sur ce poste.
- Cliquez sur le bouton **Edit Schema** et répertoriez les codes des magasins dans une colonne et les ventes dans une autre.
- Renseignez le nom de la table dans le champ **Table Name**.
- Saisissez la requête à exécuter en veillant à ce que l'ordre des colonnes correspondent à celui fourni dans le schéma. Dans cet exemple, comme toutes les colonnes du schéma ont été sélectionnées, il est possible d'utiliser l'astérisque.
- De la même façon, cliquez sur le deuxième composant pour le définir.
- Indiquez quel est le séparateur de champs : il s'agit ici d'une barre verticale.
- Passez sur la vue **Run** et cliquez sur le bouton **Run** pour exécuter le Job.

La base de données est alors analysée. Les données recherchées sont extraites et sont affichées directement sur la console (vue **Run**), où vous pouvez les voir.

Scénario 2 : Utiliser la variable StoreSQLQuery

La variable StoreSQLQuery peut être utilisée pour déboguer un scénario **tDBInput** qui ne s'exécuterait pas correctement. Elle permet d'alimenter la requête de votre composant **tDBInput** de manière dynamique.

- Partez du scénario 1 décrit ci-dessus, auquel vous ajouterez un composant **tJava**.
- Reliez les composants **tDBInput** et **tJava** via une connexion Trigger de type **OnComponentOk**. Dans le cas présent, on veut que la commande du **tDBInput** s'exécute avant la commande du **tJava**.

- Configurez les composants **tDBInput** et **tLogRow** comme dans le scénario 1 décrit plus haut.
- Passez à la vue appelée **Context**.
- Créez un nouveau paramètre que vous appellerez **StoreSQLQuery** et donnez-lui une valeur de 1 par défaut. Cette valeur indique que StoreSQLQuery est valide (**true**) et pourra être utilisé par la variable globale QUERY.
- Cliquez sur le composant **tJava** et affichez la vue **Component**. Tapez la commande `System.Out.println("")` pour afficher le contenu de la requête, puis, dans les parenthèses, appuyez sur **Ctrl+Espace** pour accéder à la liste des variables, et sélectionnez la variable globale QUERY.

- Affichez la vue **Run** et exécutez les Jobs.
- La requête associée au composant **tDBInput** s'affiche dans la console (vue **Run**), à la fin des résultats du Job :

```
971|9143650232
978|1475829234
989|3139722244
SELECT * FROM   comprehensive
Job tDBInputodbc ended at 10:15 04/03/2010. [exit code=0]
```

Pour un exemple d'utilisation du schéma dynamique dans les composants **Input**, consultez :

- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478*

tDBOutput

Propriétés du tDBOutput

Famille de composant	Databases	
Fonction	Le composant **tDBOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tDBOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent. Préférez toujours un composant Output spécifique au composant générique.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Connection type	Liste des pilotes disponibles.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*
	Clear data in table	Cochez cette case pour que les données soient supprimées de la table sélectionnée avant toute action.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.

Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de transaction (cependant pas de rollback) et surtout une meilleure performance des exécutions.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tDBOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénario : Afficher la base de données en sortie

Le Job suivant, formé de trois composants, permet de créer une nouvelle table dans une base de donnée déterminée et de l'alimenter. Le composant **tFileInputdelimited** passe le flux entrant au composant **tDBoutput**. En outre, le composant tLogRow est utilisé ici pour afficher les résultats directement dans la console de la vue **Run**.

- Cliquez et déposez les trois composants nécessaires pour ce Job.
- Définissez les paramètres d'entrée pour le composant tFileInputDelimited, dans l'onglet **Basic settings**. Dans cet exemple, le fichier contient les identifiants des propriétaires, les références des marques, des couleurs et des immatriculations des voitures. Ces informations sont organisées comme suit : un point-virgule comme séparateur de champ et un retour chariot comme séparateur de ligne. Le fichier d'entrée comprend une ligne d'en-tête à prendre en compte dans le schéma. Si ce fichier est déjà décrit dans vos métadonnées, vous pouvez en récupérer les propriétés en sélectionnant l'entrée correspondante dans le **Repository**.

- Si votre schéma est déjà stocké dans le **Repository**, sélectionnez l'option **Repository** dans le champ **Schema** puis choisissez les métadonnées appropriées à partir de la liste. Si vous n'avez encore défini aucun schéma, renseignez la structure des données dans un schéma que vous créerez en mode built-in.
- Pour ce cas de figure, réduisez l'extraction à 10 lignes.
- Définissez ensuite le composant **tDBOutput** pour configurer le flux de sortie. Sélectionnez la base de données à laquelle se connecter. Vous pouvez stocker l'ensemble des détails de connexion aux bases de données via différentes variables de contexte. Pour plus d'informations sur la manière de créer et d'utiliser les variables de contexte, consultez *Définir les variables à partir de la vue Contexts*.

- Renseignez le nom de la table dans le champ **Table**, puis choisissez les opérations à effectuer :
- Dans le champ **Action on table**, sélectionnez l'option **Drop and create table** à partir de la liste, ce qui vous permettra d'écraser la table, si elle existe, avec les nouvelles données choisies. Vous pouvez également ajouter des lignes supplémentaires dans une table déjà existante, mais notez que la gestion des doublons n'est pas supportée nativement. Consultez la section *Propriétés du tUniqRow, page 173* pour plus d'informations.
- Dans le champ **Action on data**, sélectionnez l'option **Insert**. Le flux de données entrant sera alors ajouté à la table sélectionnée.
- Pour visualiser plus facilement le flux sortant, connectez le composant tDBOuput à un composant tLogRow. Définissez le symbole barre verticale comme séparateur de champ. Appuyez sur **F6** pour exécuter le Job.
- Le traitement peut prendre un peu de temps avant que le **tLogRow** n'affiche le résultat. Pour afficher le suivi du traitement, activez la fonction **Statistics** sur la console de la vue **Run**.

Sujet associé : *Propriétés du tMysqlOutput, page 489*

tDBSQLRow

Propriétés du tDBSQLRow

Famille de composant	Databases/DB Generic	
Fonction	**tDBSQLRow** est le composant générique à tout type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie. Pour une question de performances, préférez toujours un composant de base de données spécifique au composant générique.	
Objectif	Selon la nature de la requête et de la base de données, **tDBSQLRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Datasource	Nom de la source de données ODBC à créer au préalable pour pouvoir se connecter à la base de données Pour plus d'informations, consultez le prérequis de ce tableau.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table de base de données à modifier.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.

		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	
	Utilisez le composant DBRow correspondant au type de base de données dont vous vous servez. La plupart des bases de données ont leur propre composant DBRow spécifique.	
Prérequis	ODBC (Open DataBase Connectivity) est un protocole standard permettant d'accéder aux informations de serveurs de bases de données. Vous devez préalablement créer une source de données (datasource) ODBC et la relier à votre base de données pour en lire les informations. Pour créer une source ODBC sous Windows, par exemple, allez à **Panneau de configuration** > **Outils d'administration** > **Sources de données (OBCD)** puis renseignez les propriétés de la source à partir de l'assistant.	

Scénario : Réinitialiser l'auto-incrémentation d'une base de données

Le Job suivant, à un seul composant, permet de réinitialiser l'auto-incrémentation de la base de données (valeur initiale = 1). Il ne permet pas de générer un flux en sortie car il est généralement utilisé avant l'exécution d'un script.

- Glissez-déposez le composant **tDBSQLRow** sur l'espace graphique de modélisation.
- Dans la vue **Basic settings**, renseignez les propriétés de la connexion à la source de données (datasource) ODBC.

Pour plus d'informations sur les sources de données OBCD, consultez la section concernant les prérequis dans le tableau des propriétés.

- Dans le champ Property Type, sélectionnez **Repository** pour récupérer les informations connexion stockées dans le **Repository**.
- Dans le champ **Datasource**, saisissez le nom de la source de données ODBC à laquelle est reliée la base de données contenant la table dont vous voulez réinitialiser l'auto-incrémentation.
- Dans les champs **Username** et **Password**, renseignez si nécessaire vos identifiant et mot de passe de connexion à la source de données.
- Dans le champ **Schema Type**, sélectionnez le mode **Built-in** et décrivez la structure de la base de données Talend. Le schéma n'est pas important dans notre cas particulier puisque vous souhaitez réinitialiser l'auto-incrémentation de la table et que vous ne touchez pas aux données.
- Dans le champ Table Name, renseignez le nom de la table dont vous voulez réinitialiser l'auto-incrémentation.
- Dans le champ **Query type,** laissez l'option Built-in. Cliquez sur le bouton **[...]** à côté du champ Edit schema pour ouvrir le SQLbuilder. Sinon, saisissez le code suivant :
  ```
  Alter table <TableName> auto_increment = 1
  ```
  directement dans le champ Query.

- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

L'auto-incrémentation de la base de données est réinitialisée.

Voir également : *Propriétés du tMysqlRow, page 524*.

tEXAInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tEXAInput

Famille de composant	Databases/EXA	
Fonction	Le composant **tEXAInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tEXAInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Host name	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Schema name	Saisissez le nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Saisissez le nom de la table.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
	Guess Query	Cliquez sur le bouton **Guess Query** pour générer la requête correspondant au schéma de votre table dans le champ **Query**.
	Guess schema	Cliquez sur le bouton pour récupérer le schéma de la table.
Advanced settings	*Additional JDBC parameters*	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données.
	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données EXA.	

Scénarios associés

Consultez les scénarios associés au composant **tDBInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239.*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241.*
- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478.*

tEXAOutput

Propriétés du tEXAOutput

Famille de composant	Databases/EXA	
Fonction	Le composant **tEXAOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tEXAOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro de port d'écoute du serveur.
	Schema name	Saisissez le nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Use commit control*	Cochez cette case afin de définir, dans le champ **Commit every**, qui apparaît alors, le nombre de lignes à traiter avant de commiter.

	Additional JDBC parameters	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tEXAOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données EXA. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject**s filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMysqlOutput**.

tEXARow

Propriétés du tEXARow

Famille de composant	Databases/EXA	
Fonction	**tEXARow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tEXARow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Schema name	Saisissez le nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à traiter.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.

	Guess Query	Cliquez sur le bouton **Guess Query** pour générer la requête correspondant au schéma de votre table dans le champ **Query**.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC parameters*	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données.
	Propagate QUERY's recordset	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tEXistConnection

Propriétés du tEXistConnection

Ce composant est étroitement lié aux **tEXistGet** et **tEXistPut**. Une fois que vous avez défini les propriétés de connexion du composant, vous pouvez réutiliser cette connexion sans avoir à définir à nouveau les propriétés pour chaque composant **tEXist** utilisé dans le Job.

Famille de composant	Databases/eXist	
Fonction	Le **tEXistConnection** ouvre une connexion à une base de données **eXist** afin d'effectuer une transaction.	
Objectif	Ouvrir une connexion à une base de données **eXist** afin d'effectuer une transaction.	
Basic settings	*URI*	URI de la base de données à laquelle vous souhaitez vous connecter.
	Collection	Saisissez le chemin d'accès à la collection à laquelle vous souhaitez accéder, sur le serveur de la base de données.
	Driver	Ce champ est automatiquement rempli. Par défaut, le nom du pilote standard s'affiche. Les utilisateurs peuvent spécifier un pilote différent, selon leurs besoins.
	Username et Password	Informations d'authentification de l'utilisateur de base de données.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec d'autres composants **tEXist**,comme le **tEXistGet** et le **tEXistPut**. La base de données eXist est un système de gestion de base de données open source, construit à l'aide de la technologie XML. Elle stocke des données XML selon le modèle de données XML et offre une fonction efficace de traitement de la requête XQuery, basée sur l'index. Pour plus d'informations concernant XQuery, consultez : http://exist.sourceforge.net/xquery.html (en anglais). Pour plus d'informations concernant les extensions de XQuery Update, consultez : http://exist.sourceforge.net/update_ext.html	
Limitation	n/a	

Scénario associé

Ce composant est généralement utilisé avec un **tEXistGet** et un **tEXistPut**. Il n'est pas très pertinent d'utiliser un de ces composants sans utiliser un **tEXistConnection** pour ouvrir une connexion pour la transaction.

Pour un scénario associé au **tEXistConnection**, consultez le *Scénario : Insérer des données dans des tables mère/fille, page 471*, du composant **tMysqlConnection**.

tEXistDelete

Propriétés du tEXistDelete

Famille de composant	Databases/eXist	
Fonction	Ce composant supprime des ressources d'une base de données eXist.	
Objectif	Le **tEXistDelete** supprime des ressources spécifiées dans une base de données eXist distante.	
Basic settings	*Use an existing connection/Component List*	Cochez cette case et sélectionnez le composant **tEXistConnection** adéquat à partir de la liste **Component List** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	URI	URI de la base de données à laquelle vous souhaitez vous connecter.
	Collection	Saisissez le chemin d'accès à la collection à laquelle vous souhaitez accéder, sur le serveur de la base de données.
	Driver	Ce champ est automatiquement rempli. Par défaut, le nom du pilote standard s'affiche. Les utilisateurs peuvent spécifier un pilote différent, selon leurs besoins.
	Username et Password	Informations d'authentification de l'utilisateur de base de données.
	Target Type	Sélectionnez le type de cible : **Resource**(Ressource), **Collection** (Collection), ou **All** (Tous).

	Files	Cliquez sur le bouton **[+]** afin d'ajouter les lignes que vous souhaitez utiliser comme filtres : **Filemask** : saisissez le nom du fichier ou le masque de fichier, en utilisant des caractères de remplacement (*) ou des expressions régulières.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé en tant que sous-job ou comme composant de sortie. La base de données eXist est un système de gestion de base de données open source, construit à l'aide de la technologie XML. Elle stocke des données XML selon le modèle de données XML et offre une fonction efficace de traitement de la requête XQuery, basée sur l'index. Pour plus d'informations concernant XQuery, consultez : http://exist.sourceforge.net/xquery.html (en anglais). Pour plus d'informations concernant les extensions de XQuery Update, consultez : http://exist.sourceforge.net/update_ext.html	
Limitation	n/a	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tEXistGet

tEXistGet properties

Famille de composant	Databases/eXist	
Fonction	Ce composant récupère des ressources à partir d'un serveur distant de base de données.	
Objectif	Le **tEXistGet** télécharge les ressources sélectionnées à partir d'un serveur distant de base de données dans un répertoire local.	
Basic settings	*Use an existing connection/Component List*	Cochez cette case et sélectionnez le composant **tEXistConnection** adéquat à partir de la liste **Component List** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	URI	URI de la base de données à laquelle vous souhaitez vous connecter.
	Collection	Saisissez le chemin d'accès à la collection à laquelle vous souhaitez accéder, sur le serveur de la base de données.
	Driver	Ce champ est automatiquement rempli. Par défaut, le nom du pilote standard s'affiche. Les utilisateurs peuvent spécifier un pilote différent, selon leurs besoins.
	Username et Password	Informations d'authentification de l'utilisateur de base de données.
	Local directory	Chemin d'accès au fichier de destination.

	Files	Cliquez sur le bouton **[+]** afin d'ajouter les lignes que vous souhaitez utiliser comme filtres : **Filemask** : saisissez le nom du fichier ou le masque de fichier, en utilisant des caractères de remplacement (*) ou des expressions régulières.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé en tant que sous-job ou comme composant de sortie. La base de données eXist est un système de gestion de base de données open source, construit à l'aide de la technologie XML. Elle stocke des données XML selon le modèle de données XML et offre une fonction efficace de traitement de la requête XQuery, basée sur l'index. Pour plus d'informations concernant XQuery, consultez : http://exist.sourceforge.net/xquery.html (en anglais). Pour plus d'informations concernant les extensions de XQuery Update, consultez : http://exist.sourceforge.net/update_ext.html	
Limitation	n/a	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tEXistList

Propriétés du tEXistList

Famille de composant	Databases/eXist	
Fonction	Ce composant liste les ressources stockées sur un serveur distant de base de données.	
Objectif	Le **tEXistList** liste les ressources stockées sur un serveur distant de base de données.	
Basic settings	*Use an existing connection/Component List*	Cochez cette case et sélectionnez le composant **tEXistConnection** adéquat à partir de la liste **Component List** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	URI	URI de la base de données à laquelle vous souhaitez vous connecter.
	Collection	Saisissez le chemin d'accès à la collection à laquelle vous souhaitez accéder, sur le serveur de la base de données.
	Driver	Ce champ est automatiquement rempli. Par défaut, le nom du pilote standard s'affiche. Les utilisateurs peuvent spécifier un pilote différent, selon leurs besoins.
	Username et Password	Informations d'authentification de l'utilisateur de base de données.

	Files	Cliquez sur le bouton **[+]** afin d'ajouter les lignes que vous souhaitez utiliser comme filtres : **Filemask** : saisissez le nom du fichier ou le masque de fichier, en utilisant des caractères de remplacement (*) ou des expressions régulières.
	Target Type	Sélectionnez le type de cible : **Resource**(Ressource), **Collection** (Collection), ou **All** (Tous).
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec un **tEXistGet** afin de récupérer les fichiers listés, par exemple. La base de données eXist est un système de gestion de base de données open source, construit à l'aide de la technologie XML. Elle stocke des données XML selon le modèle de données XML et offre une fonction efficace de traitement de la requête XQuery, basée sur l'index. Pour plus d'informations concernant XQuery, consultez : http://exist.sourceforge.net/xquery.html (en anglais). Pour plus d'informations concernant les extensions de XQuery Update, consultez : http://exist.sourceforge.net/update_ext.html	
Limitation	n/a	

Scénario associé

Pour un scénario associé au **tEXistList, consultez le** *Scénario : Itération sur un répertoire distant, page 1043*.

tEXistPut

Propriétés du tEXistPut

Famille de composant	Databases/eXist	
Fonction	Ce composant charge des ressources dans un serveur de base de données.	
Objectif	Le **tEXistPut** charge des fichiers spécifiés d'un répertoire local défini sur un serveur distant de base de données.	
Basic settings	*Use an existing connection/Component List*	Cochez cette case et sélectionnez le composant **tEXistConnection** adéquat à partir de la liste **Component List** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	URI	URI de la base de données à laquelle vous souhaitez vous connecter.
	Collection	Saisissez le chemin d'accès à la collection à laquelle vous souhaitez accéder, sur le serveur de la base de données.
	Driver	Ce champ est automatiquement rempli. Par défaut, le nom du pilote standard s'affiche. Les utilisateurs peuvent spécifier un pilote différent, selon leurs besoins.
	Username et Password	Informations d'authentification de l'utilisateur de base de données.
	Local directory	Chemin d'accès au fichier de destination.

	Files	Cliquez sur le bouton **[+]** afin d'ajouter les lignes que vous souhaitez utiliser comme filtres : **Filemask** : saisissez le nom du fichier ou le masque de fichier, en utilisant des caractères de remplacement (*) ou des expressions régulières.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé en tant que sous-job ou comme composant de sortie. La base de données eXist est un système de gestion de base de données open source, construit à l'aide de la technologie XML. Elle stocke des données XML selon le modèle de données XML et offre une fonction efficace de traitement de la requête XQuery, basée sur l'index. Pour plus d'informations concernant XQuery, consultez : http://exist.sourceforge.net/xquery.html (en anglais). Pour plus d'informations concernant les extensions de XQuery Update, consultez : http://exist.sourceforge.net/update_ext.html	
Limitation	n/a	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tEXistXQuery

Propriétés du tEXistXQuery

Famille de composant	Databases/eXist	
Fonction	Ce composant utilise des fichiers locaux contenant des requêtes XPath afin d'interroger des fichiers XML stockés dans une base de données distante.	
Objectif	Le **tEXistXQuery** exécute une requête sur des fichiers XML stockés sur une base de données distante, et écrit les résultats dans un fichier XML stocké localement.	
Basic settings	*Use an existing connection/Component List*	Cochez cette case et sélectionnez le composant **tEXistConnection** adéquat à partir de la liste **Component List** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	URI	URI de la base de données à laquelle vous souhaitez vous connecter.
	Collection	Saisissez le chemin d'accès à la collection à laquelle vous souhaitez accéder, sur le serveur de la base de données.
	Driver	Ce champ est automatiquement rempli. Par défaut, le nom du pilote standard s'affiche. Les utilisateurs peuvent spécifier un pilote différent, selon leurs besoins.
	Username et Password	Informations d'authentification de l'utilisateur de base de données.

	XQuery Input File	Parcourez votre répertoire jusqu'au fichier local contenant la requête à exécuter.
	Local Output	Parcourez votre répertoire jusqu'à l'emplacement où les résultats de la requête doivent être sauvegardés.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé en tant que composant unique d'un Job mais peut également être utilisé dans un Job plus complexe. La base de données eXist est un système de gestion de base de données open source, construit à l'aide de la technologie XML. Elle stocke des données XML selon le modèle de données XML et offre une fonction efficace de traitement de la requête XQuery, basée sur l'index. Pour plus d'informations concernant XQuery, consultez : http://exist.sourceforge.net/xquery.html (en anglais). Pour plus d'informations concernant les extensions de XQuery Update, consultez : http://exist.sourceforge.net/update_ext.html	
Limitation	n/a	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tEXistXUpdate

Propriétés du tEXistXUpdate

Famille de composant	Databases/eXist	
Fonction	Ce composant traite des fichiers XML et met à jour les enregistrements sur un serveur de base de données.	
Objectif	Le **tEXistXUpdate** traite des fichiers XML et met à jour les enregistrements existants sur un serveur de base de données.	
Basic settings	*Use an existing connection/Component List*	Cochez cette case et sélectionnez le composant **tEXistConnection** adéquat à partir de la liste **Component List** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	URI	URI de la base de données à laquelle vous souhaitez vous connecter.
	Collection	Saisissez le chemin d'accès à la collection à laquelle vous souhaitez accéder, sur le serveur de la base de données.
	Driver	Ce champ est automatiquement rempli. Par défaut, le nom du pilote standard s'affiche. Les utilisateurs peuvent spécifier un pilote différent, selon leurs besoins.
	Username et Password	Informations d'authentification de l'utilisateur de base de données.

	Update File	Parcourez votre répertoire jusqu'au fichier local à utiliser pour mettre à jour les enregistrements de la base de données.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé en tant que composant unique d'un Job mais peut également être utilisé dans un Job plus complexe. La base de données eXist est un système de gestion de base de données open source, construit à l'aide de la technologie XML. Elle stocke des données XML selon le modèle de données XML et offre une fonction efficace de traitement de la requête XQuery, basée sur l'index. Pour plus d'informations concernant XQuery, consultez : http://exist.sourceforge.net/xquery.html (en anglais). Pour plus d'informations concernant les extensions de XQuery Update, consultez : http://exist.sourceforge.net/update_ext.html	
Limitation	n/a	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tFirebirdClose

Propriétés du tFirebirdClose

Famille de composant	Databases/Firebird	
Fonction	Le composant **tFirebirdClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données Firebird.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tFirebirdConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tFirebirdConnection** et **tFirebirdRollback**. Il est généralement utilisé avec un composant **tFirebirdConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tFirebirdClose**, consultez *tMysqlConnection, page 471*

tFirebirdCommit

Propriétés du tFirebirdCommit

Ce composant est étroitement lié aux composants **tFirebirdConnection** et **tFirebirdRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Firebird	
Fonction	Le composant **tFirebirdCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tFirebirdConnection** dans la liste.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tFireBirdCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants Firebird et notamment avec les composants **tFirebirdConnection** et **tFirebirdRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tFirebirdConnection** et **tFirebirdRollback**. Ils sont généralement utilisés avec un composant **tFirebirdConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tFirebirdCommit**, consultez *tVerticaConnection, page 794*.

tFirebirdConnection

Propriétés du tFirebirdConnection

Ce composant est étroitement lié aux composants **tFirebirdCommit** et **tFirebirdRollback**. Ils sont généralement utilisés avec un composant **tFirebirdConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/Firebird	
Fonction	Le composant **tFirebirdConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier une fois qu'il est validé en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Table Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*Auto commit*	Cochez cette case pour que, lorsqu'une déclaration est terminée, la méthode commit soit appelée automatiquement sur cette déclaration.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants Firebird, notamment les composants **tFirebirdCommit** et **tFirebirdRollback**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au composant **tFirebirdConnection**, consultez le *Scénario : Insérer des données dans des tables mère/fille, page 471*.

tFirebirdInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tFirebirdInput

Famille de composant	Databases/Firebird	
Fonction	Le composant **tFirebirdInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tFirebirdInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.

		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Trim all the String/Char columns*	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données FireBird.	

Scénarios associés

Pour un scénario associé, consultez les scénarios associés au composant générique **tDBInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239.*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241.*
- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478.*

Consultez également le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du composant **tContextLoad**.

tFirebirdOutput

Propriétés du tFirebirdOutput

Famille de composant	Databases/Firebird	
Fonction	Le composant **tFirebirdOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tFirebirdOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro de port d'écoute du serveur.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de transaction (cependant pas de rollback) et surtout une meilleure performance des exécutions.

	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tFirebirdOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	Support null in "SQL WHERE" statement	Cochez cette case pour prendre en compte les valeurs Null d'une table de base de données. Assurez-vous que la case Nullable est bien cochée pour les colonnes du schéma correspondantes.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données Firebird. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMysqlOutput**.

tFirebirdRollback

Propriétés du tFirebirdRollback

Ce composant est étroitement lié aux composants **tFirebirdCommit** et **tFirebirdConnection**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Firebird	
Fonction	Le **tFirebirdRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant de connexion **tFirebirdConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Il faut utiliser ce composant en association avec des composants Firebird, notamment avec le **tFirebirdConnection** et le **tFirebirdCommit**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au **tFirebirdRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522* du composant **tMysqlRollback**.

tFirebirdRow

Propriétés du tFirebirdRow

Famille de composant	Databases/Firebird	
Fonction	**tFirebirdRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tFirebirdRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tFirebirdConnection** adéquat à partir de la liste **Component List** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Database	Nom de la base de données.

	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à traiter.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.

	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tGreenplumBulkExec

Propriétés du tGreenplumBulkExec

Les composants **tGreenplumOutputBulk** et **tGreenplumBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tGreenplumOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le changement des données dans la base de données.

Famille de composant	Databases/Greenplum	
Fonction	Le composant **tGreenplumBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tGreenplumBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données Greenplum.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tGreenplumConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.

	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit exister pour que l'opération d'Insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table.
	File Name	Nom du fichier à traiter.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
Advanced settings	*Action on data*	Sélectionnez l'opération que vous voulez effectuer : **Bulk insert** **Bulk update** Les informations demandées seront différentes en fonction de l'action choisie.
	Copy the OID for each row	Récupère les identifiants d'objet pour chaque ligne.
	Contains a header line with the names of each column in the file	Spécifiez que la table contient des lignes d'en-tête.
	File type	Sélectionnez le type de fichiers à traiter.
	Null string	Chaîne de caractères affichée pour signifier que la valeur est nulle.
	Fields terminated by	Caractère, chaîne ou expression régulière séparant les champs.
	Escaped char	Caractère d'échappement de la ligne.
	Text enclosure	Caractères utilisés pour entourer le texte.
	Force not null for columns	Définissez la nullabilité des colonnes **Force not null** : Cochez la case correspondant à la colonne (**Column**) que vous souhaitez définir comme n'étant pas nulle.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant est généralement utilisé avec un composant **tGreenplumOutputBulk**. Ensemble, ils offrent un gain de performance important pour l'alimentation d'une base de données Greenplum.
Limitation	n/a

Scénarios associés

Pour plus d'informations relatives au fonctionnement de **tGreenplumBulkExec**, consultez les scénarios suivants :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du composant **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du composant **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du composant **tOracleBulkExec**.

tGreenplumClose

Propriétés du tGreenplumClose

Famille de composant	Databases/Greenplum	
Fonction	Le composant **tGreenplumClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données Greenplum.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tGreenplumConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tGreenplumConnection** et **tGreenplumRollback**. Il est généralement utilisé avec un composant **tGreenplumConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tGreenplumClose**, consultez *tMysqlConnection, page 471*

tGreenplumCommit

Propriétés du tGreenplumCommit

Ce composant est étroitement lié aux composants **tGreenplumConnection** et **tGreenplumRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Greenplum	
Fonction	Le composant **tGreenplumCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tGreenplumConnection** dans la liste.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tGreenplumCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants Greenplum et notamment avec les composants **tGreenplumConnection** et **tGreenplumRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tGreenplumConnection** et **tGreenplumRollback**. Ils sont généralement utilisés avec un composant **tGreenplumConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tGreenplumCommit**, consultez *tMysqlConnection, page 471*.

tGreenplumConnection

Propriétés du tGreenplumConnection

Ce composant est étroitement lié aux composants **tGreenplumCommit** et **tGreenplumRollback**. Ils sont généralement utilisés avec un composant **tGreenplumConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/Greenplum	
Fonction	Le composant **tGreenplumConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier une fois qu'il est validé en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*Auto commit*	Cochez cette case pour que, lorsqu'une déclaration est terminée, la méthode commit soit appelée automatiquement sur cette déclaration.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Ce composant est généralement utilisé avec des composants Greenplum, notamment les composants **tGreenplumCommit** et **tGreenplumRollback**.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé au composant **tGreenplumConnection**, consultez le *Scénario : Insérer des données dans des tables mère/fille, page 471*.

tGreenplumInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tGreenplumInput

Famille de composant	Databases/ Greenplum	
Fonction	Le composant **tGreenplumInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tGreenplumInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tGreenplumConnection**.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Use cursor*	Cochez cette case et définissez le nombre de lignes avec lesquelles vous souhaitez travailler en une fois. Cette option permet d'optimiser les performances.
	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimer les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Greenplum.	

Scénarios associés

Pour un scénario associé, consultez les scénarios des composants **tDBInput** et **tMysqlInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239.*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241.*

- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478*.

Consultez également le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du composant **tContextLoad**.

tGreenplumOutput

Propriétés du tGreenplumOutput

Famille de composant	Databases/Greenplum	
Fonction	Le composant **tGreenplumOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	**Le tGreenplumOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tGreenplumConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du Guide utilisateur de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du Guide utilisateur de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.

	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tPostgresqlOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données Greenplum. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour un scénario associé au composant **tGreenplumOutput** consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du **tMysqlOutput**.

tGreenplumOutputBulk

Propriétés du tGreenplumOutputBulk

Les composants **tGreenplumOutputBulk** et **tGreenplumBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tGreenplumOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le chargement des données dans la base de données.

Famille de composant	Databases/Greenplum	
Fonction	Ecrit un fichier composé de colonnes et basé sur le délimiteur défini et sur les standards Greenplum.	
Objectif	Prépare le fichier à utiliser comme paramètre dans la requête INSERT servant à alimenter une base de données Greenplum.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
Advanced settings	*Row separator*	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Include header	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.

	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec le composant **tGreenplumBulkExec**. Ensemble, ils offrent un gain de performance important pour l'alimentation d'une base de données Greenplum.	

Scénarios associés

Pour un scénario associé au **tGreenplumOutputBulk**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du composant **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.

tGreenplumOutputBulkExec

Propriétés du tGreenplumOutputBulkExec

Les composants **tGreenplumOutputBulk** et **tGreenplumBulkExec** sont généralement utilisés ensemble comme deux parties d'un processus en deux étapes. Dans la première étape, un fichier de sortie est généré. Dans la deuxième étape, ce fichier est utilisé lors de l'opération d'INSERT afin de populer une base de données. Cette exécution en deux étapes est unifiée dans le composant **tGreenplumOutputBulkExec**.

Famille de composant	Databases/Greenplum	
Fonction	Le **tGreenplumOutputBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tGreenplumOutputBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données Greenplum.	
Basic settings	*Property Type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
Advanced settings	*Action on data*	Sélectionnez l'opération que vous voulez effectuer : **Bulk insert** **Bulk update** Les informations demandées seront différentes en fonction de l'action choisie.
	Copy the OID for each row	Récupère les identifiants d'objet pour chaque ligne.
	Contains a header line with the names of each column in the file	Spécifiez que la table contient des lignes d'en-tête.
	File type	Sélectionnez le type de fichiers à traiter.
	Null string	Chaîne de caractères affichée pour signifier que la valeur est nulle.
	Fields terminated by	Caractère, chaîne ou expression régulière séparant les champs.
	Escaped char	Caractère d'échappement de la ligne.
	Text enclosure	Caractères utilisés pour entourer le texte.
	Force not null for columns	Définissez la nullabilité des colonnes **Force not null** : Cochez la case correspondant à la colonne (**Column**) que vous souhaitez définir comme n'étant pas nulle.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est principalement utilisé lorsqu'aucune transformation particulière n'est requise sur les données à charger dans la base de données.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé au **tGreenplumOutputBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du composant **tMysqlOutputBulk**.

- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.

tGreenplumRollback

Propriétés du tGreenplumRollback

Ce composant est étroitement lié aux composants **tGreenplumCommit** et **tGreenplumConnection**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Greenplum	
Fonction	Le **tGreenplumRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant de connexion **tGreenplumConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Il faut utiliser ce composant en association avec des composants Greenplum, notamment avec le **tGreenplumConnection** et le **tGreenplumCommit**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au **tGreenplumRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522* du composant **tMysqlRollback**.

tGreenplumRow

Propriétés du tGreenplumRow

Famille de composant	Databases/Greenplum	
Fonction	Le **tGreenplumRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tGreenplumRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou **Repository**.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tGreenplumConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma

	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à lire.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Guess Query	Cliquez sur le bouton **Guess Query** pour générer la requête correspondant au schéma de votre table dans le champ **Query**.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.

	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tGreenplumSCD

Le composant **tGreenplumSCD** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tGreenplumSCD, page 7*.

tHiveClose

Propriétés du tHiveClose

Famille de composant	Databases/Hive	
Fonction	Le composant **tHiveClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données Hive.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tHiveConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié au composant **tHiveConnection**. Il est généralement utilisé avec un composant **tHiveConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tHiveClose**, consultez *tMysqlConnection, page 471*

tHiveConnection

Propriétés du tHiveConnection

Famille de composant	Databases/Hive	
Fonction	Le composant **tHiveConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier une fois qu'il est validé en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être **Built-in** ou **Repository**.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Ce composant est généralement utilisé avec des composants Hive, notamment le composant **tHiveClose**.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé au composant **tHiveConnection**, consultez le *Scénario : Insérer des données dans des tables mère/fille, page 471*.

tHiveRow

Propriétés du tHiveRow

Famille de composant	Databases/Hive	
Fonction	**tHiveRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes Hive QL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tHiveRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être **Built-in** ou **Repository**.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tHiveConnection** adéquat à partir de la liste **Component List** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.

	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à traiter.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Guess Query	Cliquez sur le bouton **Guess Query** pour générer la requête correspondant au schéma de votre table dans le champ **Query**.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes Hive QL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tHSQLDbInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tHSQLDbInput

Famille de composant	Databases/HSQLDb	
Fonction	Le composant **tHSQLDbInput** lit les données d'une base de données et extrait les champs selon la requête exécutée.	
Objectif	Le **tHSQLDbInput** exécute une requête DB dans un ordre strictement défini devant correspondre à la définition du schéma. Il passe ensuite la liste des champs au composant suivant à l'aide d'un lien **Main row**.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Running Mode	Sélectionnez dans la liste le type de serveur correspondant aux paramètres de votre base de données, parmi les quatre proposés : **HSQLDb Server**, **HSQLDb WebServer**, **HSQLDb In Process Persistent**, **HSQLDb In Memory**.
	Use TLS/SSL sockets	Cochez cette case pour autoriser, si besoin, le mode sécurisé.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.
	Database Alias	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.

	DB path	Spécifiez le chemin d'accès à la base de données à laquelle vous souhaitez vous connecter. Ce champ n'est disponible qu'en mode d'exécution **HSQLDb In Process Persistent** . Par défaut, si la base de données que vous spécifiez dans ce champ n'xiste pas, elle sera créée automatiquement. Si vous souhaitez changer la configuration par défaut, modifiez le paramètre de connexion défini dans le champ **Additional JDBC parameter**s dans la vue **Advanced settings**.
	Db name	Saisissez le nom de la base de données à laquelle vous souhaitez vous connecter. Ce champ n'est disponible qu'avec les modes d'exécution **HSQLDb In Process Persistent** et **HSQLDb In Memory**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Query type	Peut être **Built-in** ou **Repository**. **Built-in** : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder. **Repository** : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
	Guess Query	Cliquez sur le bouton **Guess Query** pour générer la requête correspondant au schéma de votre table dans le champ **Query**.

	Guess schema	Cliquez sur le bouton pour récupérer le schéma de la table.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Lorsque le mode d'exécution est **HSQLDb In Process Persistent**, cette propriété supplémentaire est définie comme `ifexists=true` par défaut, ce qui signifie que la base de données sera automatiquement créée lorsque c'est nécessaire.
	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requêtes SQL sur une base de données HSQLDb.	
Global Variables		**Number of Lines** : indique le nombre de lignes traitées. Disponible en tant que variable **After**. Retourne un entier. **Query** : indique le nombre de requêtes traitées. Disponible en tant que variable **Flow**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job* dans le **Guide Utilisateur Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : Run if, On Component Ok, On Component Error, On Subjob Ok, On Subjob Error. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate. **Trigger** : Run if, On Component Ok, On Component Error, On Subjob Ok, On Subjob Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* du **Guide Utilisateur Talend Open Studio**.

Scénarios associés

Pour des scénarios associés, consultez :

- le *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239*.

- le *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241*.
- le *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478*.

tHSQLDbOutput

Propriétés du tHSQLDbOutput

Famille de composant	Databases/HSQLDb	
Fonction	Le composant **tHSQLDbOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tHSQLDbOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Running Mode	Sélectionnez dans la liste le type de serveur correspondant aux paramètres de votre base de données, parmi les quatre proposés : **HSQLDb Server**, **HSQLDb WebServer**, **HSQLDb In Process Persistent**, **HSQLDb In Memory**.
	Use TLS/SSL sockets	Cochez cette case pour autoriser, si besoin, le mode sécurisé.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.

	DB path	Spécifiez le chemin d'accès à la base de données à laquelle vous souhaitez vous connecter. Ce champ n'est disponible qu'en mode d'exécution **HSQLDb In Process Persistent** . Par défaut, si la base de données que vous spécifiez dans ce champ n'xiste pas, elle sera créée automatiquement. Si vous souhaitez changer la configuration par défaut, modifiez le paramètre de connexion défini dans le champ **Additional JDBC parameter**s dans la vue **Advanced settings**.
	Db name	Saisissez le nom de la base de données à laquelle vous souhaitez vous connecter. Ce champ n'est disponible qu'avec les modes d'exécution **HSQLDb In Process Persistent** et **HSQLDb In Memory**.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.

Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Lorsque le mode d'exécution est **HSQLDb In Process Persistent**, cette propriété supplémentaire est définie comme `ifexists=true` par défaut, ce qui signifie que la base de données sera automatiquement créée lorsque c'est nécessaire.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitementprétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tHSQLDbOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données HSQLDb. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Global Variables		**Number of Lines** : indique le nombre de lignes traitées. Disponible en tant que variable **After**. Retourne un entier. **NB line Updated** : indique le nombre de lignes mises à jour. Disponible en tant que variable **After**. Retourne un entier. **NB line Inserted** : indique le nombre de lignes insérées. Disponible en tant que variable **After**. Retourne un entier. **NB line Deleted** : indique le nombre de lignes supprimées. Disponible en tant que variable **After**. Retourne un entier. **NB line Rejected** : indique le nombre de lignes réjetées. Disponible en tant que variable **After**. Retourne un entier. **Query** : indique le nombre de requêtes traitées. Disponible en tant que variable **Flow**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job* dans le **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : Run if, On Component Ok, On Component Error, On Subjob Ok, On Subjob Error. Liens d'entrée (d'un composant à un autre) : **Row** : Main. **Trigger** : Run if, On Component Ok, On Component Error, On Subjob Ok, On Subjob Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* du **Guide Utilisateur** **Talend Open Studio**.

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMysqlOutput**.

tHSQLDbRow

Propriétés du tHSQLDbRow

Famille de composant	Databases/HSQLDb	
Fonction	**tHSQLDbRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tHSQLDbRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Running Mode	Sélectionnez dans la liste le type de serveur correspondant aux paramètres de votre base de données, parmi les quatre proposés : **HSQLDb Server**, **HSQLDb WebServer**, **HSQLDb In Process Persistent**, **HSQLDb In Memory**.
	Use TLS/SSL sockets	Cochez cette case pour autoriser, si besoin, le mode sécurisé.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.
	Database Alias	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	DB path	Spécifiez le chemin d'accès à la base de données à laquelle vous souhaitez vous connecter. Ce champ n'est disponible qu'en mode d'exécution **HSQLDb In Process Persistent** . Par défaut, si la base de données que vous spécifiez dans ce champ n'xiste pas, elle sera créée automatiquement. Si vous souhaitez changer la configuration par défaut, modifiez le paramètre de connexion défini dans le champ **Additional JDBC parameter**s dans la vue **Advanced settings**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.

		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Guess Query	Cliquez sur le bouton **Guess Query** pour générer la requête correspondant au schéma de votre table dans le champ **Query**.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Lorsque le mode d'exécution est **HSQLDb In Process Persistent**, cette propriété supplémentaire est définie comme `ifexists=true` par défaut, ce qui signifie que la base de données sera automatiquement créée lorsque c'est nécessaire.
	Propagate QUERY's recordset	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.

	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	
Global Variables		**Query** : indique le nombre de requêtes traitées. Disponible en tant que variable **Flow**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job* dans le **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main,Reject, Iterate. **Trigger** : Run if, On Component Ok, On Component Error, On Subjob Ok, On Subjob Error. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : Run if, On Component Ok, On Component Error, On Subjob Ok, On Subjob Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* du **Guide Utilisateur** **Talend Open Studio**.

Scénarios associés

Pour des scénarios associés, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249*.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526*.

tInformixBulkExec

Propriétés du tInformixBulkExec

Les composants **tInformixOutputBulk** et **tInformixBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tInformixOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le changement des données dans la base de données.

Famille de composant	Databases/Informix	
Fonction	Le composant **tInformixBulkExec** exécute des opérations d'Insert sur les données fournies.	
Objectif	Le **tInformixBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données Informix.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Execution Platform	Sélectionnez le type de système d'exploitation que vous utilisez.

	Use an existing connection	Cochez cette case et sélectionnez le composant **tInformixConnection** adéquat à partir de la liste **Component List** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification sur l'utilisateur de base de données.
	Instance	Nom de l'instance Informix à utiliser. Cette information se trouve généralement dans le fichier *SQL hosts*.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear table** : supprimer le contenu de la table.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Informix Directory	Spécifiez le chemin d'accès à votre répertoire Informix.
	Data file	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une clé de recherche sur laquelle baser les opérations Update et Delete. Il est possible de définir les colonnes qui agiront comme clé de recherche à partir du schéma, pour une utilisation de base, ou à partir des options avancées (**Advanced settings**) pour une utilisation optimisée de ces opérations.*
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Field terminated by	Caractère, chaîne ou expression régulière séparant les champs.
	Set DBMONEY	Quand vous cochez cette case, vous pouvez définir votre séparateur décimal dans le champ **Decimal separator**.
	Set DBDATE	Sélectionnez le format de date que vous souhaitez appliquer.
	Rows Before Commit	Saisissez le nombre de lignes à traiter avant commit.

	Bad Rows Before Abort	Saisissez le nombre de lignes d'erreur avant arrêt du Job.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
	Output	Sélectionnez l'emplacement de la sortie.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes DB2.	

Scénario associé

Pour un scénario associé au composant **tInformixBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tInformixClose

Propriétés du tInformixClose

Famille de composant	Databases/Informix	
Fonction	Le composant **tInformixClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données Informix.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tInformixConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tInformixConnection** et **tInformixRollback**. Ils sont généralement utilisés avec un composant **tInformixConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tInformixClose**, consultez *tMysqlConnection, page 471*

tInformixCommit

Propriétés du tInformixCommit

Ce composant est étroitement lié aux composants **tInformixConnection** et **tInformixRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Informix	
Fonction	Le composant **tInformixCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tInformixConnection** dans la liste.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tInformixCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants Informix et notamment **tInformixConnection** et **tInformixRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tInformixConnection** et **tInformixRollback**. Ils sont généralement utilisés avec un composant **tInformixConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tInformixCommit**, consultez *tMysqlConnection, page 471*

tInformixConnection

Propriétés du tInformixConnection

Ce composant est étroitement lié aux composants **tInformixCommit** et **tInformixRollback**. Ils sont généralement utilisés avec un composant **tInformixConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/Informix	
Fonction	Le composant **tInformixConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier une fois qu'il est validé en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Instance	Nom de l'instance Informix à utiliser. Cette information se trouve généralement dans le fichier *SQL hosts*.
	Additional JDBC parameters	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*Use Transaction*	Décochez cette case lorsque la base de données est configurée en mode NO_LOG. Si la case est cochée, vous pouvez choisir d'activer ou non l'option Auto Commit.

	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants Informix, notamment les composants **tInformixCommit** et **tInformixRollback**.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé au composant **tInformixConnection**, consultez le *Scénario : Insérer des données dans des tables mère/fille, page 471*.

tInformixInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tInformixInput

Famille de composant	Databases/Informix	
Fonction	Le composant **tInformixInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tInformixInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Use an existing connection	Cochez cette case et sélectionnez le composant adéquat à partir de la liste **Component** pour réutiliser les paramètres d'une connexion que vous avez déjà définis.
	Host name	Adresse IP du serveur de base de données.
	Port	Numéro de port d'écoute du serveur.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Instance	Nom de l'instance Informix à utiliser. Cette information se trouve généralement dans le fichier *SQL hosts*.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
	Guess Query	Cliquez sur le bouton **Guess Query** pour générer la requête correspondant au schéma de votre table dans le champ **Query**.
	Guess schema	Cliquez sur le bouton pour récupérer le schéma de la table.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**..
	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Informix.	

Scénarios associés

Consultez les scénarios associés aux composants **tDBInput** et **tMysqlInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239.*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241.*
- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478*

Consultez également le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du composant **tContextLoad**.

tInformixOutput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tInformixOutput

Famille de composant	Databases/Informix	
Fonction	Le composant **tInformixOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tInformixOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Use an existing connection	Cochez cette case et sélectionnez le composant adéquat à partir de la liste **Component** pour réutiliser les paramètres d'une connexion que vous avez déjà définis.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro de port d'écoute du serveur.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Instance	Nom de l'instance Informix à utiliser. Cette information se trouve généralement dans le fichier *SQL hosts*.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.

	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).* La fonction de schéma dynamique peut être utilisée dans différents modes : **Insert**, **Update**, **Insert or Update**, **Update or insert**, **Delete**.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.

		Reference column : Saisissez une colonne de référence que le composant **tInformixOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus d'écriture dans la base de données.
	Batch size	Nombre de lignes dans chaque lot de traitement de données.
	Optimize the batch insertion	Cocher cette case optimise l'insertion de données par lots.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données Informix. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour un scénario associé au composant **tInformixOutput**, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMysqlOutput**.
- le *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507* du composant **tMysqlOutput**.

tInformixOutputBulk

Propriétés du tInformixOutputBulk

Les composants **tInformixOutputBulk** et **tInformixBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tInformixOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le chargement des données dans la base de données.

Famille de composant	Databases/Informix	
Fonction	Ecrit un fichier composé de colonnes et basé sur les délimiteur défini et sur les standards Informix.	
Objectif	Prépare le fichier à utiliser comme paramètre dans la requête INSERT servant à alimenter une base de données Informix.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
Advanced settings	*Row separator*	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Set DBMONEY	Quand vous cochez cette case, vous pouvez définir votre séparateur décimal dans le champ **Decimal separator**.

	Set DBDATE	Sélectionnez le format de date que vous souhaitez appliquer.
	Create directory if not exists	Cette case est cochée par défaut. Cette option permet de créer le dossier contenant le fichier de sortie s'il n'existe pas déjà.
	Custom the flush buffer size	Cochez cette case pour personnaliser la taille de la mémoire utilisée pour stocker temporairement les données, et dans le champ **Row number**, saisissez le nombre de lignes après lesquelles la mémoire est à nouveau libérée.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec le composant **tInformixBulkExec**. Ensemble, ils offrent un gain de performance important pour l'alimentation d'une base de données Informix.	

Scénarios associés

Pour un scénario associé au **tInformixOutputBulk**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du composant **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.

tInformixOutputBulkExec

Propriétés du tInformixOutputBulkExec

Les composants **tInformixOutputBulk** et **tInformixBulkExec** sont généralement utilisés ensemble comme deux parties d'un processus en deux étapes. Dans la première étape, un fichier de sortie est généré. Dans la deuxième étape, ce fichier est utilisé lors de l'opération d'INSERT afin de populer une base de données. Cette exécution en deux étapes est unifiée dans le composant **tInformixOutputBulkExec**.

Famille de composant	Databases/Informix	
Fonction	Le **tInformixOutputBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tInformixOutputBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données Informix.	
Basic settings	*Property Type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Execution platform	Sélectionnez le type de système d'exploitation que vous utilisez.

	Use an existing connection	Cochez cette case et sélectionnez le composant **tInformixConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Instance	Nom de l'instance Informix à utiliser. Cette information se trouve généralement dans le fichier *SQL hosts*.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear table** : supprimer le contenu de la table.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Informix Directory	Spécifiez le chemin d'accès à votre répertoire Informix.
	Data file	Nom du fichier à traiter. Voir également :*Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Action on data	Sélectionnez l'opération que vous voulez effectuer : **Bulk insert** **Bulk update** Les informations demandées seront différentes en fonction de l'action choisie.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Row separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Fields terminated by	Caractère, chaîne ou expression régulière séparant les champs.
	Set DBMONEY	Quand vous cochez cette case, vous pouvez définir votre séparateur décimal dans le champ **Decimal separator**.
	Set DBDATE	Sélectionnez le format de date que vous souhaitez appliquer.
	Rows Before Commit	Saisissez le nombre de lignes à traiter avant commit.
	Bad Rows Before Abort	Saisissez le nombre de lignes d'erreur avant arrêt du Job.
	Create directory if not exists	Cette case est cochée par défaut. Cette option permet de créer le dossier contenant le fichier de sortie s'il n'existe pas déjà.
	Custom the flush buffer size	Cochez cette case pour personnaliser la taille de la mémoire utilisée pour stocker temporairement les données, et dans le champ **Row number**, saisissez le nombre de lignes après lesquelles la mémoire est à nouveau libérée.

	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
	Output	Sélectionnez l'emplacement de la sortie.
Utilisation	Ce composant est principalement utilisé lorsqu'aucune transformation particulière n'est requise sur les données à charger dans la base de données.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé au **tInformixOutputBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du composant **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.

tInformixRollback

Propriétés du tInformixRollback

Ce composant est étroitement lié aux composants **tInformixCommit** et **tInformixConnection**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Informix	
Fonction	Le **tInformixRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant de connexion **tInformixConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Il faut utiliser ce composant en association avec des composants Informix, notamment avec le **tInformixConnection** et le **tInformixCommit**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au **tInformixRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522* du composant **tMysqlRollback**.

tInformixRow

Propriétés du tInformixRow

Famille de composant	Databases/Informix	
Fonction	**tInformixRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tInformixRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tInformixConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.

	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Instance	Nom de l'instance Informix à utiliser. Cette information se trouve généralement dans le fichier *SQL hosts*.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Guess Query	Cliquez sur le bouton **Guess Query** pour générer la requête correspondant au schéma de votre table dans le champ **Query**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
	Use Transaction	Décochez cette case lorsque la base de données est configurée en mode NO_LOG.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Propagate QUERY's recordset	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.

	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des “?” dans l’instruction SQL définie dans le champ **Query** de l’onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l’instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l’écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d’exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l’auto-incrémentation d’une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tInformixSCD

Le composant **tInformixSCD** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tInformixSCD, page 9*.

tInformixSP

Propriétés du tInformixSP

Famille de composant	Databases/Informix	
Fonction	Le composant **tInformixSP** appelle une procédure stockée de base de données.	
Objectif	Le **tInformixSP** permet de centraliser des requêtes multiples ou complexes dans une base de données et de les appeler plus facilement.	
Basic settings	*Property type*	Peut être **Built-in** ou **Repository**.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tInformixConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro de port d'écoute du serveur.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.

	Instance	Nom de l'instance Informix à utiliser. Cette information se trouve généralement dans le fichier *SQL hosts*.
	Schema et *Edit Schema*	Dans une procédure stockée, le schéma est un paramètre d'entrée. Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le **Repository**. Ainsi, il est réutilisable. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	SP Name	Saisissez le nom exact de la Procédure Stockée (SP).
	Is Function / *Return result in*	Cochez cette case si une seule valeur doit être retournée. Sélectionnez dans la liste la colonne du schéma sur laquelle est basée la valeur à obtenir.
	Parameters	Cliquez sur le bouton **[+]** et sélectionnez dans le champ **Schema Columns** les différentes colonnes nécessaires à la procédure. Notez que le schéma de la SP peut contenir plus de colonnes qu'il n'y a de paramètres utilisés dans la procédure. Sélectionnez le **Type** de paramètre : **IN** : paramètre d'entrée (Input) **OUT** : paramètre de sortie (Output)/valeur retournée **IN OUT** : les paramètres d'entrée doivent être retournées sous forme de valeur, même après modifications via la procédure (fonction). **RECORDSET** : les paramètres d'entrée doivent être retournées sous forme d'ensemble de valeurs, au lieu d'une valeur unique. Consultez le composant *tParseRecordSet* si vous voulez analyser un ensemble d'enregistrements d'une table de données ou d'une requête SQL.
	Use Transaction	Décochez cette case lorsque la base de données est configurée en mode NO_LOG.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est un composant intermédiaire. Il peut être utilisé comme composant de début. Dans ce cas, seuls les paramètres d'entrée sont autorisés.	
Limitation	La syntaxe de la Procédure Stockée doit correspondre à celle de la base de données.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Trouver le libellé State à l'aide d'une procédure stockée, page 537* du composant **tMysqlSP**.
- le *Scénario : Vérifier le format de numéros à l'aide d'une procédure stockée, page 605* du composant **tOracleSP**.

Consultez également le composant *tParseRecordSet, page 635* si vous voulez analyser un ensemble d'enregistrements d'une table de données ou d'une requête SQL.

tIngresClose

Propriétés du tIngresClose

Famille de composant	Databases/Ingres	
Fonction	Le composant **tIngresClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données Ingres.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tIngresConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tIngresConnection** et **tIngresRollback**. Ils sont généralement utilisés avec un composant **tIngresConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tIngresClose**, consultez *tMysqlConnection, page 471*

tIngresCommit

Propriétés du tIngresCommit

Ce composant est étroitement lié aux composants **tIngresConnection** et **tIngresRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Ingres	
Fonction	Le composant **tIngresCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	Sélectionnez le composant **tIngresConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Close Connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tIngresCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants JDBC et notamment avec les composants **tIngresConnection** et **tIngresRollback**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au **tIngresCommit**, consultez le *Scénario : Insérer des données dans des tables mère/fille, page 471*.

tIngresConnection

Propriétés du tIngresConnection

Ce composant est étroitement lié aux composants **tIngresCommit** et **tIngresRollback**. Ils sont généralement utilisés avec un composant **tIngresConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/Ingres	
Fonction	Le composant **tIngresConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être **Built-in** ou **Repository**.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Server	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Table Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Utilisation	Ce composant est généralement utilisé avec des composants JDBC et notamment avec les composants **tIngresCommit** et **tIngresRollback**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au **tIngresConnection**, consultez le *Scénario : Insérer des données dans des tables mère/fille, page 471*.

tIngresInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tIngresInput

Famille de composant	Databases/Ingres	
Fonction	Le composant **tIngresInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tIngresInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Server	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Trim all the String/Char columns*	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Ingres.	

Scénarios associés

Consultez les scénarios associés au composant **tDBInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239.*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241.*
- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478.*

Consultez également le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du composant **tContextLoad**.

tIngresOutput

Propriétés du tIngresOutput

Famille de composant	Databases/Ingres	
Fonction	Le composant **tIngresOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tIngresOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro de port d'écoute du serveur.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.

	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tIngresOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données Ingres. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMysqlOutput**.

tIngresRollback

Propriétés du tIngresRollback

Ce composant est étroitement lié aux composants **tIngresCommit** et **tIngresConnection**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Ingres	
Fonction	Le **tIngresRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant de connexion **tIngresConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Il faut utiliser ce composant en association avec des composants Ingres, notamment avec le **tIngresConnection** et le **tIngresCommit**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au **tIngresRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522* du composant **tMysqlRollback**.

tIngresRow

Propriétés du tIngresRow

Famille de composant	Databases/Ingres	
Fonction	**tIngresRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tIngresRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à traiter.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.

	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tIngresSCD

Le composant **tIngresSCD** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tIngresSCD, page 11*.

tInterbaseClose

Propriétés du tInterbaseClose

Famille de composant	Databases/Interbase	
Fonction	Le composant **tInterbaseClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données Interbase.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tInterbaseConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tInterbaseConnection** et **tInterbaseRollback**. Ils sont généralement utilisés avec un composant **tInterbaseConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tInterbaseClose**, consultez *tMysqlConnection, page 471*

tInterbaseCommit

Propriétés du tInterbaseCommit

Ce composant est étroitement lié aux composants **tInterbaseConnection** et **tInterbaseRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Interbase	
Fonction	Le composant **tInterbaseCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	Sélectionnez le composant **tInterbaseConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Close Connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tInterbaseCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants Interbase et notamment avec les composants **tInterbaseConnection** et **tInterbaseRollback**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au **tInterbaseCommit**, consultez le *Scénario : Insérer des données dans des tables mère/fille, page 471*.

tInterbaseConnection

Propriétés du tInterbaseConnection

Ce composant est étroitement lié aux composants **tInterbaseCommit** et **tInterbaseRollback**. Ils sont généralement utilisés avec un composant **tInterbaseConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/Interbase	
Fonction	Le composant **tInterbaseConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier une fois qu'il est validé en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Table Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Utilisation	Ce composant est généralement utilisé avec des composants Interbase, notamment les composants **tInterbaseCommit** et **tInterbaseRollback**.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé au composant **tInterbaseConnection**, consultez le *Scénario : Insérer des données dans des tables mère/fille, page 471*.

tInterbaseInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tInterbaseInput

Famille de composant	Databases/Interbase	
Fonction	Le composant **tInterbaseInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tInterbaseInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Host	Adresse IP du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.

		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Trim all the String/Char columns*	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Interbase.	

Scénarios associés

Consultez les scénarios associés au composant **tDBInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239.*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241.*
- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478.*

Consultez également le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* associé au composant **tContextLoad**.

tInterbaseOutput

Propriétés du tInterbaseOutput

Famille de composant	Databases/Interbase	
Fonction	Le composant **tInterbaseOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tInterbaseOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Host	Adresse IP du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*
	Clear data in table	Cochez cette case pour que les données soient supprimées de la table sélectionnée avant toute action.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.

Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tInterbaseOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données Interbase. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* associé au composant **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* associé au composant **tMysqlOutput**.

tInterbaseRollback

Propriétés du tInterbaseRollback

Ce composant est étroitement lié aux composants **tInterbaseCommit** et **tInterbaseConnection**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Interbase	
Fonction	Le **tInterbaseRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant de connexion **tInterbaseConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Il faut utiliser ce composant en association avec des composants Interbase, notamment avec le **tInterbaseConnection** et le **tInterbaseCommit**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au **tInterbaseRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522* du composant **tMySQLRollback**.

tInterbaseRow

Propriétés du tInterbaseRow

Famille de composant	Databases/Interbase	
Fonction	**tInterbaseRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tInterbaseRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tInterbaseConnection** adéquat à partir de la liste **Component List** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Database	Nom de la base de données.

	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table Name	Nom de la table à traiter.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.

	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249* associé au composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* associé au composant **tMySQLRow**.

tJavaDBInput

Propriétés du tJavaDBInput

Famille de composant	Databases/JavaDB	
Fonction	Le composant **tJavaDBInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tJavaDBInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Framework	Sélectionnez dans la liste un framework pour votre base de données Java.
	Database	Nom de la base de données.
	DB root path	Parcourez vos fichiers jusqu'à la racine de votre base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.

		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Query type et Query	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Trim all the String/Char columns*	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données JavaDB.	

Scénarios associés

Pour un scénario associé, consultez les scénarios du composant **tDBInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239.*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241.*
- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478.*

Consultez également le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du composant **tContextLoad**.

tJavaDBOutput

Propriétés du tJavaDBOutput

Famille de composant	Databases/JavaDB	
Fonction	Le composant **tJavaDBOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tJavaDBOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Framework	Sélectionnez dans la liste un framework pour votre base de données Java.
	Database	Nom de la base de données.
	DB root path	Parcourez vos fichiers jusqu'à la racine de votre base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.

	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tJavaDBOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données JavaDB. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios asoociés

Pour un scénario associé, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMysqlOutput**.

tJavaDBRow

Propriétés du tJavaDBRow

Famille de composant	Databases/JavaDB	
Fonction	**tJavaDBRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tJavaDBRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Framework Type	Sélectionnez dans la liste un framework pour votre base de données Java.
	Database	Nom de la base de données.
	DB root path	Parcourez vos fichiers jusqu'à la racine de votre base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à traiter.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.

		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tJDBCColumnList

Propriétés du tJDBCColumnList

Famille de composants	Databases/JDBC	
Fonction	Le composant **tJDBCColumList** effectue une opération d'itération dans toutes les colonnes d'une table donnée, grâce à une connexion JDBC définie.	
Objectif	Le **tJDBCColumList** permet de faire une liste des libellés des colonnes d'une table JDBC donnée.	
Basic settings	*Component list*	Sélectionnez le composant **tJDBCConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Table name	Nom de la table à lire.
Utilisation	Il faut utiliser ce composant en association avec les autres composants JDBC, notamment avec le **tJDBCConnection**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Itérer une table de base de données et lister le nom des colonnes de la table, page 466*.

tJDBCClose

Propriétés du tJDBCClose

Famille de composant	Databases/JDBC	
Fonction	Le composant **tJDBCClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données JDBC.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tJDBCConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tJDBCConnection** et **tJDBCRollback**. Il est généralement utilisé avec un composant **tJDBCConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tJDBCClose**, consultez *tMysqlConnection, page 471*

tJDBCCommit

Propriétés du tJDBCCommit

Ce composant est étroitement lié aux composants **tJDBCConnection** et **tJDBCRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/JDBC	
Fonction	Le composant **tJDBCCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	Sélectionnez le composant **tJDBCConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tJDBCCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants JDBC et notamment avec les composants **tJDBCConnection** et **tJDBCRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tJDBCConnection** et **tJDBCRollback**. Ils sont généralement utilisés avec un composant **tJDBCConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tJDBCCommit**, consultez *tMysqlConnection, page 471*.

tJDBCConnection

Propriétés du tJDBCConnection

Ce composant est étroitement lié aux composants **tJDBCCommit** et **tJDBCRollback**. Ils sont généralement utilisés avec un composant **tJDBCConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/JDBC	
Fonction	Le composant **tJDBCConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers une base de données de sortie.	
Basic settings		
	JDBC URL	Entrez l'URL JDBC dans le champ associé pour vous connecter à la base de données. Par exemple, entrez *jdbc:mysql://IP address/database name* pour vous connecter à une base de données mysql.
	Driver JAR	Cliquez sur le bouton **[+]** sous le tableau afin d'ajouter autant de lignes que nécessaire pour charger différents JARs. Dans chaque ligne, cliquez ensuite sur le bouton **[...]** pour ouvrir l'assistant **Select Module**, dans lequel vous pouvez sélectionner le pilote JAR de votre choix pour chaque ligne.
	Driver Class	Renseignez la classe du driver dans le champ **Driver Class** en fonction de votre connexion. Par exemple, entrez *com.mysql.jdbc.Driver* dans ce champ pour vous connecter à une base de données mysql.
	Username et *Password*	Entrez les informations d'authentification de l'utilisateur de base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*Use Auto commit*	Cochez cette case afin d'afficher la case **Auto Commit**. Si vous la cochez, elle activera le mode auto-commit. Une fois la case **Use Auto-Commit** décochée, la case **Auto-Commit** ne sera plus visible.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log, au niveau de chaque composant.
Utilisation	Ce composant est généralement utilisé avec des composants JDBC et notamment avec les composants **tJDBCCommit** et **tJDBCRollback**.	

Limitation	n/a

Scénario associé

Ce composant est étroitement lié aux composants **tJDBCCommit** et **tJDBCRollback**. Ils sont généralement utilisés avec un composant **tJDBCConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tJDBCConnection**, consultez *tMysqlConnection, page 471*.

tJDBCInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tJDBCInput

Famille de composant	Databases/JDBC	
Fonction	Le composant **tJDBCInput** lit toute base de données via une API JDBC et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tJDBCInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tJDBCConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.

		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	JDBC URL	Entrez le chemin d'accès à la base de données.
	Driver JAR	Cliquez sur le bouton **[+]** sous le tableau afin d'ajouter autant de lignes que nécessaire pour charger différents JARs. Dans chaque ligne, cliquez ensuite sur le bouton **[...]** pour ouvrir l'assistant **Select Module**, dans lequel vous pouvez sélectionner le pilote JAR de votre choix pour chaque ligne.
	Class Name	Dans le champ **Class name, renseignez le nom de la classe pour qu'elle soit indiquée dans le driver.**
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à lire.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*

Advanced settings	*Use cursor*	Cochez cette case et définissez le nombre de lignes avec lesquelles vous souhaitez travailler en une fois. Cette option permet d'optimiser les performances.
	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données utilisant une connection JDBC.	

Scénarios associés

Pour un scénario associé, consultez les scénarios du composant **tDBInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239.*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241.*
- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478.*

Consultez également le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du composant **tContextLoad**.

tJDBCOutput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tJDBCOutput

Famille de composant	Databases/JDBC	
Fonction	Le composant **tJDBCOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tJDBCOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut-être **Built-in** ou **Repository**.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tJDBCConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.

		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	JDBC URL	Entrez le chemin d'accès à la base de données.
	Driver JAR	Cliquez sur le bouton **[+]** sous le tableau afin d'ajouter autant de lignes que nécessaire pour charger différents JARs. Dans chaque ligne, cliquez ensuite sur le bouton **[...]** pour ouvrir l'assistant **Select Module**, dans lequel vous pouvez sélectionner le pilote JAR de votre choix pour chaque ligne.
	Class Name	Dans le champ **Class name, renseignez le nom de la classe pour qu'elle soit indiquée dans le driver.**
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tJDBCOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.

	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus d'écriture dans la base de données.
	Use Batch Size	Nombre de lignes dans chaque lot de traitement de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données JDBC. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour un scénario associé au **tJDBCOutput**, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMysqlOutput**.

tJDBCRollback

Propriétés du tJDBCRollback

Ce composant est étroitement lié aux composants **tJDBCCommit** et **tJDBCConnection**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/JDBC	
Fonction	Le **tJDBCRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant de connexion **tJDBCConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Close connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Il faut utiliser ce composant en association avec des composants JDBC, notamment avec le **tJDBCConnection** et le **tJDBCCommit**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tJDBCConnection** et **tJDBCCommit**. Ils sont généralement utilisés avec un composant **tJDBCConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tJDBCRollback**, consultez *tMysqlConnection, page 471*.

tJDBCRow

Propriétés du tJDBCRow

Famille de composant	Databases/JDBC	
Fonction	**tJDBCRow** est le composant générique pour tout type de base de données utilisant une API JDBC. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tJDBCRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Use an existing connection*	Cochez cette case et sélectionnez le composant **tJDBCConnection** adéquat à partir de la liste **Component** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	JDBC URL	Entrez le chemin d'accès à la base de données.
	Driver JAR	Cliquez sur le bouton **[+]** sous le tableau afin d'ajouter autant de lignes que nécessaire pour charger différents JARs. Dans chaque ligne, cliquez ensuite sur le bouton **[...]** pour ouvrir l'assistant **Select Module**, dans lequel vous pouvez sélectionner le pilote JAR de votre choix pour chaque ligne.
	Class Name	Renseignez le nom de la classe pour qu'elle soit indiquée dans le driver.

	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.

	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur toute base de données utilisant une connexion JDBC et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tJDBCSP

Propriétés du tJDBCSP

Famille de composant	Databases/JDBC	
Fonction	Le composant **tJDBCSP** appelle une procédure stockée de base de données.	
Objectif	Le **tJDBCSP** permet de centraliser des requêtes multiples ou complexes dans une base de données et de les appeler plus facilement.	
Basic settings	*JDBC URL*	Entrez le chemin d'accès à la base de données.
	Driver JAR	Cliquez sur le bouton **[+]** sous le tableau afin d'ajouter autant de lignes que nécessaire pour charger différents JARs. Dans chaque ligne, cliquez ensuite sur le bouton **[...]** pour ouvrir l'assistant **Select Module**, dans lequel vous pouvez sélectionner le pilote JAR de votre choix pour chaque ligne.
	Class Name	Dans le champ **Class name**, renseignez le nom de la classe pour qu'elle soit indiquée dans le driver.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Dans une procédure stockée, le schéma est un paramètre d'entrée. Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le **Repository**. Ainsi, il est réutilisable. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	SP Name	Saisissez le nom exact de la Procédure Stockée (SP).
	Is Function / *Return result in*	Cochez cette case si une seule valeur doit être retournée. Sélectionnez dans la liste la colonne du schéma sur laquelle est basée la valeur à obtenir.

	Parameters	Cliquez sur le bouton (+) et sélectionnez dans le champ **Schema Columns** les différentes colonnes nécessaires à la procédure. Notez que le schéma de la SP peut contenir plus de colonnes qu'il n'y a de paramètres utilisés dans la procédure. Sélectionnez le **Type** de paramètre : **IN** : paramètre d'entrée (Input) **OUT** : paramètre de sortie (Output)/valeur retournée **IN OUT** : les paramètres d'entrée doivent être retournées sous forme de valeur, même après modifications via la procédure (fonction). **RECORDSET** : les paramètres d'entrée doivent être retournées sous forme d'ensemble de valeurs, au lieu d'une valeur unique. Consultez le composant *tParseRecordSet* si vous voulez analyser un ensemble d'enregistrements d'une table de données ou d'une requête SQL.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est un composant intermédiaire. Il peut être utilisé comme composant de début. Dans ce cas, seuls les paramètres d'entrée sont autorisés.	
Limitation	La syntaxe de la Procédure Stockée doit correspondre à celle de la base de données.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Trouver le libellé State à l'aide d'une procédure stockée, page 537* du composant **tMysqlSP**.
- le *Scénario : Vérifier le format de numéros à l'aide d'une procédure stockée, page 605* du composant **tOracleSP**.

Consultez également le composant *tParseRecordSet, page 635* si vous voulez analyser un ensemble d'enregistrements d'une table de données ou d'une requête SQL.

tJDBCTableList

Propriétés du tJDBCTableList

Famille de composant	Databases/JDBC	
Fonction	Le composant **tJDBCTableList** effectue une opération d'itération sur toutes les tables d'une base de données, grâce à une connexion JDBC définie.	
Objectif	Le composant **tJDBCTableList** permet de lister les noms des tables JDBC grâce aux commandes SELECT et WHERE.	
Basic settings	*Component list*	Sélectionnez le composant de connexion **tJDBCConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Where clause for table name selection	Saisissez la commande WHERE permettant d'identifier les tables sur lesquelles effectuer l'opération d'itération.
Utilisation	Il faut utiliser ce composant en association avec les autres composants JDBC, notamment avec le **tJDBCConnection**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Itérer une table de base de données et lister le nom des colonnes de la table, page 466*.

tLDAPAttributesInput

Propriétés du tLDAPAttributesInput

Famille de composant	Databases/LDAP	
Fonction	Le composant **tLDAPAttributesInput** analyse chaque élément trouvé via une requête LDAP et liste une série d'attributs associés à l'objet.	
Objectif	Le **tLDAPAttributesInput** exécute une LDAP basée sur un filtre prédéfini et correspondant au schéma défini. Puis la liste de champ est transmise au composant suivant via une connexion **Main row**.	
Basic settings	*Property type*	Peut-être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur d'annuaire LDAP.
	Port	Numéro du port d'écoute du serveur.
	Base DN	Chemin d'accès à l'arborescence de l'utilisateur autorisé.
	Protocol	Sélectionnez le type de protocole dans la liste. **LDAP** : aucun codage n'est utilisé **LDAPS** : LDAP sécurisé. Quand cette option est sélectionnée, la case **Advanced CA** apparaît. Une fois sélectionné, le mode avancé vous permet de spécifier le répertoire et le mot de de passe de la clé secrète du certificat pour le stockage d'un CA spécifique. Cependant, vous pouvez toujours désactiver la validation du certificat, en cochant la case **Trust all certs**. **TLS** : le certificat est utilisé. Quand cette option est sélectionnée, la case **Advanced CA** apparaît et est utilisée de la même façon que pour le type **LPDAPS**.
	Authentication User et *Password*	Cochez **Authentication** si une connexion LDAP est nécessaire. Notez que le login doit correspondre à la syntaxe LDAP pour être valide. Par exemple : "cn=Directory Manager".
	Filter	Saisissez le filtre comme attendu par l'annuaire LDAP db.
	Multi valued field separator	Saisissez le séparateur de valeur des champs à valeurs multiples.

	Alias dereferencing	Sélectionnez l'option dans la liste. **Never** permet l'amélioration des performances de recherche si vous êtes sûr qu'aucun alias n'est déréférencé. Par défaut, utilisez **Always** : **Always** : les alias sont toujours déréférencés. **Never** : les alias ne sont jamais déréférencés. **Searching** : cette option déréférence les alias uniquement après la résolution du nom. **Finding** : cette option déréférence les alias uniquement lors de la résolution du nom.
	Referral handling	Sélectionnez l'option dans la liste : **Ignore** : ne tient pas compte des redirections des requêtes **Follow** : tient compte des redirections des requêtes
	Limit	Cette option permet de limiter le nombre d'enregistrement lus, si nécessaire.
	Time Limit	Cette option permet de limiter la durée de connexion à l'annuaire.
	Paging	Spécifiez le nombre d'entrées retournées en une fois par le serveur LDAP.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**. *Comme ce composant sert à lister les attributs associés à un élément LDAP, sont schéma est prédéfini. Conservez ces colonnes définies, même si vous avez besoin d'ajouter de nouvelles colonnes. Pour cela, utilisez le mode Built-In.*
		Le schéma prédéfini comprend : - *objectclass* : liste des classes d'éléments - *mandatoryattributes* : liste des attributs obligatoires pour ces classes - *optionalattributes* : liste des attributs facultatifs pour ces classes - *objectattributes* : liste des attributs essentiels aux éléments analysés.

Advanced settings	*Class Definition Root*	Spécifiez la racine de l'espace de nom de l'élément Class definition.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les requêtes LDAP possibles. Note : Appuyez sur **Ctrl + Espace** pour accéder à la liste des variables globales, inclus la variable **GetResultName** permettant de récupérer automatiquement la base correspondante.	

Scénario associé

Le composant **tLDAPAttributesInput** fonctionne de manière similaire au **tLDAPInput**. Vous pouvez donc consulter le *Scénario : Afficher le contenu filtré d'un annuaire LDAP, page 414*

tLDAPInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tLDAPInput

Famille de composant	Databases/LDAP	
Fonction	Le composant **tLDAPInput** lit un répertoire et extrait les données en fonction d'un filtre défini.	
Objectif	Le **tLDAPInput** exécute une LDAP basée sur un filtre prédéfini et correspondant au schéma défini. Puis la liste de champ est transmise au composant suivant via une connexion **Main row**.	
Basic settings	*Property type*	Peut-être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio** .
	Host	Adresse IP du serveur d'annuaire LDAP.
	Port	Numéro du port d'écoute du serveur.
	Base DN	Chemin d'accès à l'arborescence de l'utilisateur autorisé. Pour retrouver les informations complètes du Distinguished Name, ajoutez un champ appellé *DN* dans le schéma, en minuscules ou en majuscules.
	Protocol	Sélectionnez le type de protocole dans la liste. **LDAP** : aucun codage n'est utilisé **LDAPS** : LDAP sécurisé. Quand cette option est sélectionnée, la case **Advanced CA** apparaît. Une fois sélectionné, le mode avancé vous permet de spécifier le répertoire et le mot de de passe de la clé secrète du certificat pour le stockage d'un CA spécifique. Cependant, vous pouvez toujours désactiver la validation du certificat, en cochant la case **Trust all certs**. **TLS** : le certificat est utilisé. Quand cette option est sélectionnée, la case **Advanced CA** apparaît et est utilisée de la même façon que pour le type **LPDAPS**.

	Authentication User et *Password*	Cochez **Authentication** si une connexion LDAP est nécessaire. Notez que le login doit correspondre à la syntaxe LDAP pour être valide. Par exemple : “cn=Directory Manager”.
	Filter	Saisissez le filtre comme attendu par l’annuaire LDAP db.
	Multi valued field separator	Saisissez le séparateur de valeur des champs à valeurs multiples.
	Alias dereferencing	Sélectionnez l’option dans la liste. **Never** permet l’amélioration des performances de recherche si vous êtes sûr qu’aucun alias n’est déréférencé. Par défaut, utilisez **Always** : **Always** : les alias sont toujours déréférencés. **Never** : les alias ne sont jamais déréférencés. **Searching** : cette option déréférence les alias uniquement après la résolution du nom. **Finding** : cette option déréférence les alias uniquement lors de la résolution du nom.
	Referral handling	Sélectionnez l’option dans la liste : **Ignore** : ne tient pas compte des redirections des requêtes **Follow** : tient compte des redirections des requêtes
	Limit	Cette option permet de limiter le nombre d’enregistrement lus, si nécessaire.
	Time Limit	Cette option permet de limiter la durée de connexion à l’annuaire.
	Paging	Spécifiez le nombre d’entrées retournées en une fois par le serveur LDAP.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d’erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d’une source sans avoir à mapper chaque colonne individuellement. Pour plus d’informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.

		Repository : Le schéma existe déjà et il est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
Utilisation	Ce composant couvre toutes les requêtes LDAP possibles. Note : Appuyez sur **Ctrl + Espace** pour accéder à la liste des variables globales, inclus la variable **GetResultName** permettant de récupérer automatiquement la base correspondante.	

Scénario : Afficher le contenu filtré d'un annuaire LDAP

Le Job décrit ci-dessous filtre simplement l'annuaire LDAP et affiche le résultat dans la console Run Job.

- Cliquez et déposez les composants **tLDAPInput** et **tLogRow**.
- Paramétrez les propriétés du **tLDAPInput**.
- Sélectionnez **Repository** dans le champ **Property type** si vous avez stocké les informations de connexion LDAP dans le **Metadata Manager** du **Repository**. Puis sélectionnez l'entrée adéquate dans la liste.
- En mode **Built-In**, renseignez manuellement les champs **Host** et **Port**. Host peut être l'adresse IP du serveur de l'annuaire LDAP ou son nom DNS.
- Aucune **Base DN** particulière ne doit être paramétrée.

- Puis dans le champ **Protocol**, sélectionnez le protocole adéquat dans la liste. Dans cet exemple : un protocole **LDAP** simple est utilisé.
- Cochez la case **Authentication** et renseignez les informations de connexion pour lire l'annuaire, si nécessaire. Dans ce scénario, aucune authentification n'est nécessaire.
- Dans la zone **Filter**, saisissez la commande sur laquelle est basée la sélection de données. Dans cet exemple, le filtre est : `(&(objectClass=inetorgperson)&(uid=PIERRE DUPONT))`.
- Renseignez le champ **Multi-valued field separator** avec une virgule, puisque certains champs contiennent plus d'une valeur séparée par une virgule.
- Comme vous ne savez pas si des alias sont utilisés dans l'annuaire LDAP, sélectionnez **Always** dans la liste.
- Sélectionnez **Ignore** dans le champ **Referral handling**.
- Dans le champ **Limit**, définissez la limite à **100** dans cet exemple.

tLDAPInput_1

Column	Db Column	Key	Type	Nullable	Date P...	Length	Pre...
dc	dc	☐	String	☑		255	
ou	ou	☐	String	☑		255	
objectClass	objectClass	☐	String	☑		255	
mail	mail	☐	String	☑		255	
uid	uid	☐	String	☑		255	
dn	dn	☐	String	☐			

- Définissez le **Schéma** tel qu'il est requis par l'annuaire LDAP. Dans cet exemple, le schéma est composé de 6 colonnes dont les colonnes `objectClass` et `uid` qui seront filtrées.
- Dans le composant **tLogRow**, aucune configuration particulière n'est exigée.

```
Starting job testLDAPInput at 18:05 18/09/2007.
DATA|top,person,organizationalPerson,inetorgperson,a4400user|mhirt78@talend.com|PIERRE DUPONT|
Job testLDAPInput ended at 18:05 18/09/2007. [exit code=0]
```

Seule une entrée de l'annuaire correspond aux critères du filtre défini dans le composant **tLDAPInput**.

Pour un exemple d'utilisation du schéma dynamique dans les composants **Input**, consultez :

- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478*

tLDAPOutput

Propriétés du tLDAPOutput

Famille de composant	Databases/LDAP	
Fonction	Le composant **tLDAPOutput** écrit dans un annuaire LDAP.	
Objectif	Le **tLDAPOutput** exécute une requête LDAP basée sur un filtre prédéfini et correspondant à un schéma défini. Puis la liste de champ est transmise au composant suivant via une connexion **Main row**.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Host	Adresse IP du serveur d'annuaire LDAP.
	Port	Numéro de port d'écoute du serveur.
	Base DN	Chemin d'accès à l'arborescence de l'utilisateur autorisé.
	Protocol	Sélectionnez le type de protocole dans la liste. **LDAP** : aucun codage n'est utilisé **LDAPS** : LDAP sécurisé. Quand cette option est sélectionnée, la case **Advanced CA** apparaît. Une fois sélectionné, le mode avancé vous permet de spécifier le répertoire et le mot de de passe de la clé secrète du certificat pour le stockage d'un CA spécifique. Cependant, vous pouvez toujours désactiver la validation du certificat, en cochant la case **Trust all certs**. **TLS** : le certificat est utilisé. Quand cette option est sélectionnée, la case **Advanced CA** apparaît et est utilisée de la même façon que pour le type **LPDAPS**.
	User et *Password*	Renseignez les champs **User** (utilisateur) et **Password** (mot de passe) conformément à l'annuaire. Notez que le login doit correspondre à la syntaxe LDAP pour être valide. Par exemple : "cn=Directory Manager".
	Multi valued field separator	Caractère, chaîne ou expression régulière séparant les informations dans un champ multi-valué.

	Alias dereferencing	Sélectionnez l'option dans la liste. **Never** permet l'amélioration des performances de recherche si vous êtes sûr qu'aucun alias n'est déréférencé. Par défaut, utilisez **Always**. **Always** : les alias sont toujours déréférencés. **Never** : les alias ne sont jamais déréférencés. **Searching** : cette option déréférence les alias uniquement après la résolution du nom. **Finding** : cette option déréférence les alias uniquement lors de la résolution du nom.
	Referral handling	Sélectionnez l'option dans la liste : **Ignore** : ne tient pas compte des redirections des requêtes. **Follow** : tient compte des redirections des requêtes.
	Insert mode	Sélectionnez l'action dans la liste : **Add** : ajoute une valeur dans un attribut multi-valué **Insert** : insère de nouvelles données **Update** : met à jour les données existantes **Delete** : supprime les données sélectionnées de l'annuaire **Insert or Update** : insère et met à jour les données
	DN Column Name	Sélectionnez, à partir de la liste déroulante, le type d'entrée LDAP utilisé.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les requêtes LDAP possibles. Note : Appuyez sur **Ctrl + Espace** pour accéder à la liste des variables globales, inclus la variable **GetResultName** permettant de récupérer automatiquement la base correspondante. Ce composant permet de faire des actions sur une table ou les données d'une table d'une base de données Oracle. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénario : Editer des données dans un annuaire LDAP

Le scénario suivant décrit un Job qui lit un annuaire LDAP, met à jour l'email d'une entrée sélectionnée et affiche la sortie avant de mettre à jour l'annuaire LDAP en sortie. C'est-à-dire qu'aucun alias n'est déréférencé et qu'aucune requête n'est redirigée. Ce scénario est basé sur le scénario du **LDAPInput** (cf. *Scénario : Afficher le contenu filtré d'un annuaire LDAP, page 414*). Le résultat obtenu est une entrée simple, correspondant à une personne organisationnelle dont l'email est mis à jour.

- Cliquez et déposez les composants **tLDAPInput**, **tLDAPOutput**, **tMap** et **tLogRow** dans l'espace de modélisation.
- Connectez le composant d'entrée au **tMap** puis au **tLogRow** puis au composant de sortie.
- Dans la vue **Component** du **tLDAPInput**, paramétrez les informations de connexion au serveur d'annuaire LDAP, ainsi que les filtres tel que décrit dans le *Scénario : Afficher le contenu filtré d'un annuaire LDAP, page 414*.
- Simplifiez le schéma en enlevant les champs suivants : *dc*, *ou*, *objectclass*.

- Puis ouvrez le mapper pour définir les changements à exécuter.
- Glissez et déposez la colonne *uid* de la table d'entrée (input) à la table de sortie (output) puisque aucune modification n'est nécessaire pour cette colonne.

- Dans le champ **Expression** de la colonne *dn* de la table de sortie (output), saisissez l'expression exacte attendue par le serveur LDAP afin qu'il atteigne l'arborescence souhaitée et permette d'écrire dans l'annuaire, à condition que vous n'ayez pas déjà renseigné le champ **Base DN** du composant **tLDAPOutput**.

- Dans cet exemple, la variable globale **GetResultName** est utilisée pour récupérer automatiquement le chemin d'accès. Appuyez sur **Ctrl+Espace** pour accéder à la liste de variable et sélectionnez `tLDAPInput_1_RESULT_NAME`.
- Dans le champ **Expression** de la colonne *mail*, saisissez l'adresse email qui remplacera celle obsolète dans l'annuaire LDAP. Dans cet exemple, le nouvel email est : *Pierre.Dupont@talend.com*.
- Cliquez sur **OK** pour valider les modifications.
- Le composant **tLogRow** ne nécessite pas de paramétrage particulier.
- Puis sélectionnez le composant **tLDAPOutput** pour définir les propriétés d'écriture dans l'annuaire.

- Définissez manuellement les champs **Port** et **Host** si ces informations ne sont pas stockées dans le **Repository**.
- Dans le champ **Base DN**, définissez la plus haute arborescence à laquelle vous avez accès. Si vous n'avez pas défini le chemin complet et exact au DN cible, définissez-le ici. Dans cet exemple, le DN complet est fournit par la sortie *dn* du composant **tMap**, Ainsi, seule la plus haute arborescence accessible est fournie : *o=directoryRoot*.
- Sélectionnez le protocole à utiliser : **LDAP** pour cet exemple.
- Puis renseignez les champs **User** et **Password** attendus par l'annuaire LDAP.
- Laissez les paramètres par défaut des champs **Alias Dereferencing** et **Referral Handling**, c'est-à-dire respectivement **Always** et **Ignore**.
- Dans le champ **Insert mode**, sélectionnez l'option **Update** pour cet exemple (l'adresse email).
- Le schéma est récupéré du composant précédent à l'aide d'une opération de propagation.
- Enregistrez le Job et exécutez-le en appuyant sur la touche **F6**.

```
Starting job LDAPInputnew at 14:17 20/09/2007.
uid=PIERRE DUPONT,ou=DATA,o=TALENDM,o=TALEND|PIERRE DUPONT|Pierre.Dupont@talend.com
Job LDAPInputnew ended at 14:17 20/09/2007. [exit code=0]
```

La sortie montre les champs suivants : *dn*, *uid* et *mail* comme défini dans le Job.

tLDAPRenameEntry

Propriétés du tLDAPRenameEntry

Famille de composant	Databases/LDAP	
Fonction	Le composant **tLDAPRenameEntry** renomme des entrées dans un annuaire LDAP.	
Objectif	Le **tLDAPRenameEntry** permet de renommer une ou des entrée(s) dans un annuaire LDAP sélectionné.	
Basic settings	*Property type*	Peut-être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur d'annuaire LDAP.
	Port	Numéro du port d'écoute du serveur.
	Base DN	Chemin d'accès à l'arborescence de l'utilisateur autorisé.
	Protocol	Sélectionnez le type de protocole dans la liste. **LDAP** : aucun codage n'est utilisé **LDAPS** : LDAP sécurisé **TLS** : le certificat est utilisé
	User et *Password*	Renseignez les informations d'authentification de l'utilisateur. Notez que le login doit correspondre à la syntaxe LDAP pour être valide. Par exemple : "cn=Directory Manager".
	Alias dereferencing	Sélectionnez l'option dans la liste. **Never** permet l'amélioration des performances de recherche si vous êtes sûr qu'aucun alias n'est déréférencé. Par défaut, utilisez **Always** : **Always** : les alias sont toujours déréférencés. **Never** : les alias ne sont jamais déréférencés. **Searching** : cette option déréférence les alias uniquement après la résolution du nom. **Finding** : cette option déréférence les alias uniquement lors de la résolution du nom.
	Referrals handling	Sélectionnez l'option dans la liste : **Ignore** : ne tient pas compte des redirections des requêtes **Follow** : tient compte des redirections des requêtes
	Previous DN et *New DN*	Sélectionnez à partir de la liste déroulante la colonne du schéma portant l'ancien DN (**Previous DN**) et celle portant le nouveau DN à appliquer (**New DN**).

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
Utilisation	Ce composant couvre toutes les requêtes LDAP possibles. Il est généralement utilisé comme sous-Job à un seul composant mais il peut aussi être utilisé en association avec d'autres composants. Note : Appuyez sur **Ctrl + Espace** pour accéder à la liste des variables globales, inclus la variable **GetResultName** permettant de récupérer automatiquement la base correspondante.	

Scénarios associés

Pour un scénario associé au composant **tLDAPRenameEntry**, consultez :

- le *Scénario : Afficher le contenu filtré d'un annuaire LDAP, page 414* du composant **tLDAPInput**.
- le *Scénario : Editer des données dans un annuaire LDAP, page 419* du **tLDAPOutput**.

tMaxDBInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tMaxDBInput

Famille de composant	Databases/MaxDB	
Fonction	Le composant **tMaxDBInput** lit une table de base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tMaxDBInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio** .
	Host name	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.

		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table name	Entrez le nom de la table.
	Query type et Query	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
	Guess Query	Cliquez sur le bouton **Guess Query** pour générer la requête correspondant au schéma de votre table dans le champ **Query**.
	Guess schema	Cliquez sur le bouton pour récupérer le schéma de la table.
Advanced settings	*Trim all the String/Char columns*	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez les scénarios :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239.*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241.*
- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478.*

tMaxDBOutput

Propriétés du tMaxDBOutput

Famille de composant	Databases/MaxDB	
Fonction	Le composant **tMaxDBOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tMaxDBOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.

	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tMaxDBOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données MaxDB. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMysqlOutput**.

tMaxDBRow

Propriétés du tMaxDBRow

Famille de composant	Databases/MaxDB0	
Fonction	**tMaxDBRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tMaxDBRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table name	Entrez le nom de la table.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le Repository. Le champ **Query** est renseigné automatiquement.

	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tMSSqlBulkExec

Propriétés du tMSSqlBulkExec

Les composants **tMSSqlOutputBulk** et **tMSSqlBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tMSSqlOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le changement des données dans la base de données.

Famille de composant	Databases/MSSql	
Fonction	Effectue une action d'Insert sur les données fournies.	
Objectif	Le composant **tMSSqlBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données MSSql.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tMSSqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.

	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit exister pour que l'opération d'Insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Remote File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Action*	Sélectionnez l'opération que vous voulez effectuer : **Bulk insert** **Bulk update** **Bcp query out** Les informations demandées seront différentes en fonction de l'action choisie.
Bulk insert & Bulk update	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Fields terminated	Caractère, chaîne ou expression régulière séparant les champs.
	Rows terminated	Caractère, chaîne ou expression régulière séparant les lignes.

	First row	Saisissez le numéro du rang à partir duquel vous voulez démarrer l'action.
	Code page	Cette valeur peut prendre plusieurs formes : **OEM** (valeur par défaut) **ACP** **RAW** Personnalisée par l'utilisateur
	Data file type	Sélectionnez le type de données à traiter.
	Output	Sélectionnez la sortie à utiliser pour récupérer la sortie standard de la base MSSql : **to console** : vers la console. **to global variable** : vers la variable globale.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Bcp query out	*Fields terminated*	Caractère, chaîne ou expression régulière séparant les champs.
	Rows terminated	Caractère, chaîne ou expression régulière séparant les lignes.
	Data file type	Sélectionnez le type de données à traiter.
	Output	Sélectionnez la sortie à utiliser pour récupérer la sortie standard de la base MSSql : **to console** : vers la console. **to global variable** : vers la variable globale.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec un composant **tMSSqlOutputBulk**. Ensemble, ils offrent un gain de performance important pour l'alimentation d'une base de données MSSql.	

Scénarios associés

Pour un scénario associé au composant **tMSSqlBulkExec**, consultez les scénarios suivants :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.

tMSSqlClose

Propriétés du tMSSqlClose

Famille de composant	Databases/MSSql	
Fonction	Le composant **tMSSqlClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données MSSql.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tMSSqlConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tMSSqlConnection** et **tMSSqlRollback**. Il est généralement utilisé avec un composant **tMSSqlConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tMSSqlClose**, consultez *tMysqlConnection, page 471*

tMSSqlColumnList

Propriétés du tMSSqlColumnList

Famille de composant	Databases/MS SQL	
Fonction	Le composant **tMSSqlColumList** effectue une opération d'itération dans toutes les colonnes d'une table donnée, grâce à une connexion MS SQL définie.	
Objectif	Le **tMSSqlColumList** permet de faire une liste des libellés des colonnes d'une table MS SQL donnée.	
Basic settings	*Component list*	Sélectionnez le composant **tMSSqlConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Table name	Entrez le nom de la table.
Utilisation	Il faut utiliser ce composant en association avec les autres composants MS SQL, notamment avec le **tMSSqlConnection**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Itérer une table de base de données et lister le nom des colonnes de la table, page 466*.

tMSSqlCommit

Propriétés du tMSSqlCommit

Ce composant est étroitement lié aux composants **tMSSqlConnection** et **tMSSqlRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/MSSql	
Fonction	Le composant **tMSSqlCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tMSSqlConnection** dans la liste.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tMSSqlCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants MSSql et notamment avec les composants **tMSSqlConnection** et **tMSSqlRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tMSSqlConnection** et **tMSSqlRollback**. Ils sont généralement utilisés avec un composant **tMSSqlConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tMSSqlCommit**, consultez *tMysqlConnection, page 471*.

tMSSqlConnection

Propriétés du tMSSqlConnection

Ce composant est étroitement lié aux composants **tMSSqlCommit** et **tMSSqlRollback**. Ils sont généralemet utilisés avec un composant **tMSSqlConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/MSSQL	
Fonction	Le composant **tMSSqlConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Schema	Nom du schéma.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Additional JDBC parameters	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*Auto commit*	Cochez cette case pour que, lorsqu'une déclaration est terminée, la méthode commit soit appelée automatiquement sur cette déclaration.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log, aussi bien au niveau du Job qu'au niveau de chaque composant.

Utilisation	Ce composant est généralement utilisé avec des composants MSSql, notamment les composants **tMSSqlCommit** et **tMSSqlRollback**.
Limitation	n/a

Scénario associé

Ce composant est étroitement lié aux composants **tMSSqlCommit** et **tMSSqlRollback**. Ils sont généralement utilisés avec un composant **tMSSqlConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tMSSqlConnection**, consultez *tMysqlConnection, page 471*.

tMSSqlInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tMSSqlInput

Famille de composant	Databases/MS SQL Server	
Fonction	Le composant **tMSSqlInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tMSSqlInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Use an existing connection*	Cochez cette case lorsque vous utilisez le composant **tMSSqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Property type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.

		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant (Repository).
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table name	Nom de la table à lire.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.

	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données MS Sql.	

Scénarios associés

Pour un scénario associé, consultez les scénarios des composants **tDBInput** et **tMysqlInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239.*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241.*
- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478*

Consultez également le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du **tContextLoad**.

tMSSqlLastInsertId

Propriétés du tMSSqlLastInsertId

Famille de composant	Databases/MS SQL server	
Fonction	**tMSSqlLastInsertId** affiche les dernières ID ajoutées à une table à partir d'une connexion MSSql spécifiée.	
Objectif	**tMSSqlLastInsertId** permet de récupérer les clés primaires des entrées dernièrement ajoutées par un utilisateur à une table MSSql.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Component list	Sélectionnez le composant **tMSSqlConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL.	

Scénario associé

Pour un scénario associé au composant **tMSSqlLastInsertId**, consultez *Scénario : Récupérer les ID des dernières entrées ajoutées, page 484*.

tMSSqlOutput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tMSSqlOutput

Famille de composant	Databases/MS SQL server	
Fonction	Le composant **tMSSqlOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tMSSqlOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut-être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tMSSqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Turn on identity insert	Cochez cette case pour utiliser votre propre séquence sur les valeurs Identity des données insérées (plutôt que de laisser le serveur SQL choisir les valeurs séquentielles).

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée.. **Insert if not exist** : Ajouter de nouvelles entrées à la table si nécessaire. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).* La fonction de schéma dynamique peut être utilisée dans différents modes : **Insert**, **Update**, **Insert or Update**, **Update or insert**, **Delete**.
	Specify identity field	Cochez cette case pour personnaliser l'incrémentation : **Identity field** : sélectionnez la colonne à incrémenter. **Start value** : définissez la valeur d'incrémentation de départ. **Step** : définissez la valeur à incrémenter à chaque ligne.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.

		Reference column : Saisissez une colonne de référence que le composant **tMSSqlOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus d'écriture dans la base de données.
	Support null in "SQL WHERE" statement	Cochez cette case pour prendre en compte les valeurs Null d'une table de base de données. Assurez-vous que la case Nullable est bien cochée pour les colonnes du schéma correspondantes.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
	Use Batch Size	Nombre de lignes dans chaque lot de traitement de données.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données MSSql. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour un scénario associé au composant **tMSSqlOutput**, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMysqlOutput**.
- *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507* du composant **tMysqlOutput**.

tMSSqlOutputBulk

Propriétés du tMSSqlOutputBulk

Les composants **tMSSqlOutputBulk** et **tMSSqlBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tMSSqlOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le chargement des données dans la base de données.

Famille de composant	Databases/MSSql	
Fonction	Ecrit un fichier composé de colonnes et basé sur le délimiteur défini et sur les standards MSSql.	
Objectif	Prépare le fichier à utiliser comme paramètre dans la requête INSERT servant à alimenter une base de données MSSQL.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
Advanced settings	*Row separator*	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Include header	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.

	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec le composant **tMSSQlBulkExec**. Ensemble, ils offrent un gain de performance important pour l'alimentation d'une base de données MSSQL.	

Scénarios associés

Pour un scénario associé au **tMSSqlOutputBulk**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du composant **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.

tMSSqlOutputBulkExec

Propriétés du tMSSqlOutputBulkExec

Les composants **tMSSqlOutputBulk** et **tMSSqlBulkExec** sont généralement utilisés ensemble comme deux parties d'un processus en deux étapes. Dans la première étape, un fichier de sortie est généré. Dans la deuxième étape, ce fichier est utilisé lors de l'opération d'INSERT afin de populer une base de données. Cette exécution en deux étapes est unifiée dans le composant **tMSSqlOutputBulkExec**.

Famille de composant	Databases/MSSql	
Fonction	Le **tMSSqlOutputBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tMSSqlOutputBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données MSSQL.	
Basic settings	*Action*	Permet de choisir entre les options **Bulk Insert** et **Bulk update**.
	Property Type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tMSSqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.

	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**..
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Row separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	First row	Saisissez le numéro du rang à partir duquel vous voulez démarrer l'action.

	Include header	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.
	Data file type	Sélectionnez le type de données à traiter.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est principalement utilisé lorsqu'aucune transformation particulière n'est requise sur les données à charger dans la base de données.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé au **tMSSqlOutputBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du composant **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.

tMSSqlRollback

Propriétés du tMSSqlRollback

Ce composant est étroitement lié aux composants **tMSSqlCommit** et **tMSSqlConnection**. Ils sont généralement utilisés ensemble lors d'une transaction.

Famille de composant	Databases	
Fonction	Le **tMSSqlRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant **tMSSqlConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce component est généralement utilisé avec d'autres composants MSSql, notamment les composants **tMSSqlConnection** et **tMSSqlCommit** components.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522*.

tMSSqlRow

Propriétés du tMSSqlRow

Famille de composant	Databases/DB2	
Fonction	**tMSSqlRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tMSSqlRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Use an existing connection*	Cochez cette case lorsque vous utilisez le composant **tMSSqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Property type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.

	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit exister pour que l'opération d'Insert soit autorisée.
	Turn on identity insert	Cochez cette case pour utiliser votre propre séquence sur les valeurs Identity des données insérées (plutôt que de laisser le serveur SQL choisir les valeurs séquentielles).
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Propagate QUERY's recordset	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.

	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des “?” dans l’instruction SQL définie dans le champ **Query** de l’onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l’instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l’écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d’exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l’auto-incrémentation d’une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du **tMySQLRow**.

tMSSqlSCD

Le composant **tMSSqlSCD** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tMSSqlSCD, page 17*.

tMSSqlSP

Propriétés du tMSSqlSP

Famille de composant	Databases/MSSql	
Fonction	Le composant **tMSSqlSP** appelle une procédure stockée de base de données.	
Objectif	Le **tMSSqlSP** permet de centraliser des requêtes multiples ou complexes dans une base de données et de les appeler plus facilement.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier des propriétés du composant. Les champs suivants sont alors pré-remplis.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tMSSqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification sur l'utilisateur de base de données.

	Schema et *Edit Schema*	Dans une procédure stockée, le schéma est un paramètre d'entrée. Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le **Repository**. Ainsi, il est réutilisable. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	SP Name	Saisissez le nom exact de la Procédure Stockée (SP).
	Is Function / Return result in	Cochez cette case, si une seule valeur doit être retournée. Sélectionnez dans la liste la colonne du schéma sur laquelle est basée la valeur à obtenir.
	Parameters	Cliquez sur le bouton (+) et sélectionnez dans le champ **Schema Columns** les différentes colonnes nécessaires à la procédure. Notez que le schéma de la SP peut contenir plus de colonnes qu'il n'y a de paramètres utilisés dans la procédure. Sélectionnez le **Type** de paramètre : **IN** : paramètre d'entrée (Input) **OUT** : paramètre de sortie (Output)/valeur retournée **IN OUT** : les paramètres d'entrée doivent être retournées sous forme de valeur, même après modifications via la procédure (fonction). **RECORDSET** : les paramètres d'entrée doivent être retournées sous forme d'ensemble de valeurs, au lieu d'une valeur unique. Consultez le composant *tParseRecordSet* si vous voulez analyser un ensemble d'enregistrements d'une table de données ou d'une requête SQL.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est un composant intermédiaire. Il peut être utilisé comme composant de début. Dans ce cas, seuls les paramètres d'entrée sont autorisés.	
Limitation	La syntaxe de la Procédure Stockée doit correspondre à celle de la base de données.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Trouver le libellé State à l'aide d'une procédure stockée, page 537* du composant **tMysqlSP**.
- le *Scénario : Vérifier le format de numéros à l'aide d'une procédure stockée, page 605* du **tOracleSP**.

Consultez également le composant *tParseRecordSet, page 635* si vous voulez analyser un ensemble d'enregistrements d'une table de données ou d'une requête SQL.

tMSSqlTableList

Propriétés du tMSSqlTableList

Famille de composant	Databases/MS SQL	
Fonction	Le composant **tMSSqlTableList** effectue une opération d'itération sur toutes les tables d'une base de données, grâce à une connexion MS SQL définie.	
Objectif	Le composant **tMSSqlTableList** permet de lister les noms des tables MS SQL grâce aux commandes SELECT et WHERE.	
Basic settings	*Component list*	Sélectionnez le composant **tMSSqlConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours..
	Where clause for table name selection	Saisissez la commande WHERE permettant d'identifier les tables sur lesquelles effectuer l'opération d'itération.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Il faut utiliser ce composant en association avec les autres composants MS SQL, notamment avec le **tMSSqlConnection**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Itérer une table de base de données et lister le nom des colonnes de la table, page 466*.

tMysqlBulkExec

Propriétés du tMysqlBulkExec

Les composants **tMysqlOutputBulk** et **tMysqlBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tMysqlOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le changement des données dans la base de données.

Famille de composant	Databases/Mysql	
Fonction	Le composant **tMysqlBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tMysqlBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données MySQL.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	DB Version	Sélectionnez la version de MySQL que vous utilisez.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tMysqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.

	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune action sur la table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Table	Nom de la table à écrire. Une seule table peut être écrite à la fois et cette table doit déjà exister pour que l'opération d'Insert soit autorisée.
	Local File Name	Chemin d'accès et nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
Advanced settings	*Additional JDBC parameters*	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données.
	Lines terminated by	Caractère ou suite de caractères utilisés pour séparer les lignes.
	Fields terminated by	Caractère, chaîne ou expression régulière séparant les champs.
	Enclosed by	Caractères entourant les données.
	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert records in table** : Insère de nouveaux enregistrements dans une table. **Replace records in table** : Remplace les enregistrements existants par de nouveaux. **Update records in table** : Modifie les enregistrements existants. **Ignore records in table** : Ignore les enregistrements existants ou insère les nouveaux.

	Records contain NULL value	Cochez cette case si vous souhaitez récupérer les valeurs nulles du flux d'entrée. Si vous ne cochez pas cette case, les valeurs nulles en entrée seront considérées comme des champs vides dans le flux de sortie.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec un composant **tMysqlOutputBulk**. Ensemble, ils offrent un gain de performance important pour l'alimentation d'une base de données MySQL.	
Limitation	n/a	

Scénarios associés

Pour plus d'informations relatives au fonctionnement de **tMysqlBulkExec**, consultez les scénarios suivants :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du composant **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du composant **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du composant **tOracleBulkExec**.

tMysqlClose

Propriétés du tMysqlClose

Famille de composant	Databases/Mysql	
Fonction	Le composant **tMysqlClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données Mysql.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tMysqlConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tMysqlConnection** et **tMysqlRollback**. Il est généralement utilisé avec un composant **tMysqlConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tMysqlClose**, consultez *tMysqlConnection, page 471*

tMysqlColumnList

Propriétés du tMysqlColumnList

Famille de composants	Databases/MySQL	
Fonction	Le composant **tMysqlColumList** effectue une opération d'itération dans toutes les colonnes d'une table donnée, grâce à une connexion MySQL définie.	
Purpose	Le **tMysqlColumList** permet de faire une liste des libellés des colonnes d'une table MySQL donnée.	
Basic settings	*Component list*	Sélectionnez le composant **tMysqlConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Table name	Entrez le nom de la table.
Usage	Il faut utiliser ce composant en association avec les autres composants MySQL, notamment avec le **tMysqlConnection**.	
Limitation	n/a	

Scénario : Itérer une table de base de données et lister le nom des colonnes de la table

Le scénario Java suivant permet de créer un Job à cinq composants. Ce Job permet d'itérer une table donnée à partir d'une base de données MySQL en utilisant la commande Where et de lister tous les libellés des colonnes de la table.

- Dans la palette, cliquez sur chacun des composants suivants et déposez-les dans l'espace de modélisation : **tMysqlConnection**, **tMysqlTableList**, **tMysqlColumnList**, **tFixedFlowInput** et **tLogRow**.
- Reliez le **tMysqlConnection** au **tMysqlTableList** par le biais d'un lien **OnSubJobOk**.
- Reliez les composants **tMysqlTableList**, **tMysqlColumnList** et **tFixedFlowInput** à l'aide de liens **Iterate**.
- Reliez le **tFixedFlowInput** au **tLogRow** par le biais d'un lien **Row Main**.

- Dans l'espace de modélisation, sélectionnez le composant **tMysqlConnection** et cliquez sur la vue **Component** pour définir les paramètres de base.
- Dans l'onglet **Basic settings**, entrez manuellement les propriétés de connexion à la base de données ou sélectionnez-les à partir de la liste des variables de contexte : appuyez simultanément sur les touches **Ctrl+Espace** puis cliquez sur le champ correspondant si vous avez stocké les informations en local sous le noeud **DB Connections** du répertoire **Metadata**.

Pour plus d'informations sur les métadonnées, veuillez consulter la section *Centraliser les métadonnées* du **Guide Utilisateur** de Talend Open Studio.

Dans cet exemple, on veut se connecter à une base de données MySQL appelée *customers*.

- Dans l'espace de modélisation, sélectionnez le composant **tMysqlTableList** et cliquez sur la vue **Component** pour définir ses paramètres de base.

- Dans le champ **Component list**, sélectionnez le composant de connexion MySQL adéquat si vous utilisez plus d'une connexion.
- Renseignez la condition Where en utilisant la bonne syntaxe dans le champ correspondant pour itérer les noms de la ou des tables que vous voulez lister dans la console.
  Dans ce scénario, on veut itérer la table appelée *customer*.
- Dans l'espace de modélisation, sélectionnez le composant **tMysqlColumnList** et cliquez sur la vue **Component** pour paramétrer ses propriétés de base.

- Dans le champ **Component list**, sélectionnez le composant de connexion MySQL adéquat si vous utilisez plus d'une connexion.
- Dans le champ **Table name**, saisissez le nom de la table dont vous souhaitez lister les libellés de colonnes.
  Dans ce scénario, l'objectif est de lister les libellés des colonnes de la table *customer*.
- Dans l'espace de modélisation, sélectionnez le composant **tFixedFlowInput** et cliquez sur la vue **Component** pour paramétrer ses propriétés de base.
- Dans le champ **Schema**, sélectionnez l'option **Built-In** et cliquez sur le bouton **[...]** à côté du champ **Edit Schema** pour définir les données que vous souhaitez utiliser en entrée. Dans ce scénario, le schéma est composé de deux colonnes : la première *TableName* correspond au nom de la table MySQL et la deuxième *ColumnName* correspond aux libellés des colonnes.

- Cliquez sur **OK** pour fermer la boîte de dialogue et une nouvelle boîte de dialogue apparaît vous demandant si vous souhaitez propager ces modifications, acceptez en cliquant sur **Yes**. Les colonnes définies apparaissent dans le champ **Values** de la vue **Basic settings**.
- Dans le champ **Values**, cliquez sur chacune des cellules de la colonne **Value** pour définir le contenu des colonnes de la table MySQL et appuyez sur **Ctrl+Espace** pour accéder à la liste des variables globales.

- Dans la liste des variables globales, selectionnez : *((String)globalMap.get("tMysqlTableList_1_CURRENT_TABLE"))* pour la colonne *TableName* et *((String)globalMap.get("tMysqlColumnList_1_COLUMN_NAME"))* pour la colonne *ColumnName*.

- Dans l'espace de modélisation, sélectionnez le composant **tLogRow**.
- Cliquez sur la vue **Component** et paramétrez les propriétés de base du **tLogRow** en fonction de vos besoins.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job Column_Table_List at 00:55 17/11/2008.
customer|id
customer|First_Name
customer|Last_Name
customer|Address
customer|id_State
Job Column_Table_List ended at 00:55 17/11/2008. [exit code=0]
```

Le nom de la table MySQL et le libellé de ses colonnes sont affichés dans la console.

tMysqlCommit

Propriétés du tMysqlCommit

Ce composant est étroitement lié aux composants **tMysqlConnection** et **tMysqlRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/MySQL	
Fonction	Le composant **tMysqlCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tMysqlConnection** dans la liste.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tMysqlCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants MySQL et notamment avec les composants **tMysqlConnection** et **tMysqlRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tMysqlConnection** et **tMysqlRollback**. Ils sont généralement utilisés avec un composant **tMysqlConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tMysqlCommit**, consultez *tMysqlConnection, page 471*.

tMysqlConnection

Propriétés du tMysqlConnection

Ce composant est étroitement lié aux composants **tMysqlCommit** et **tMysqlRollback**. Ils sont généralemet utilisés avec un composant **tMysqlConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/MySQL	
Fonction	Le composant **tMysqlConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Table Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Utilisation	Ce composant est généralement utilisé avec des composants MySQL, notamment les composants **tMysqlCommit** et **tMysqlRollback**.	
Limitation	n/a	

Scénario : Insérer des données dans des tables mère/fille

Le Job suivant est dédié à des utilisateurs avancés de base de données qui souhaitent effectuer des insertions dans des tables multiples à l'aide de l'*id* d'une table parent qui alimentera une table enfant.

Avant de créer ce Job, suivez les étapes décrites ci-dessous pour créer les tables nécessaires, à l'aide d'un gestionnaire de table tel que *innodb*.

- Dans un éditeur de ligne de commande, connectez-vous à votre serveur MySQL.
- Une fois connecté à la base de données correspondante, saisissez la commande suivante afin de créer la table parent : `create table f1090_mum(id int not null auto_increment, name varchar(10), primary key(id)) engine=innodb;`
- Puis créez la deuxième table : `create table baby (id_baby int not null, years int) engine=innodb;`

Dans **Talend Open Studio**, le Job est constitué de sept composants, **tMysqlConnection** et **tMysqlCommit** inclus.

- Cliquez et déposez les composants suivants de la **Palette** : **tFileList**, **tFileInputDelimited**, **tMap**, **tMysqlOutput** (x2).
- Connectez le composant **tFileList** au composant d'entrée **tFileInputDelimited** à l'aide d'une connexion de type **Iterate**. Ainsi, le nom du fichier à traiter sera automatiquement renseigné à partir de la liste de fichiers du **tFileList** en utilisant une variable globale.
- Connectez le composant **tFileInputDelimited** au **tMap** et séparez le flux vers deux composants de sortie **tMysqlOutput**. Utilisez des connexions de type **Row** pour chacune de ces connexions représentant un flux de données principal.
- Définissez les propriétés du composant **tFileList**, notamment le répertoire dans lequel sont stockés les fichiers.
- Ajoutez le composant **tMysqlConnection** et connectez-le au composant de départ de ce Job, (dans cet exemple, le composant **tFileList**) à l'aide d'une connexion **OnComponentOk** afin de définir l'ordre d'exécution.
- Dans la vue **Component** du composant **tMysqlConnection**, définissez manuellement les informations de connexion ou récupérez-les dans le **Repository** si vous les avez préalablement stockées dans le dossier **DB connection** du répertoire **Metadata**. Pour plus d'information concernant les Métadonnées, consultez *Centraliser les métadonnées* du **Guide Utilisateur** de **Talend Open Studio**.

- Dans l'onglet **Basic settings** du composant **tFileInputDelimited**, appuyez sur la touche **Ctrl+Espace** pour accéder à la liste des variables. Définissez le champ **File Name** avec la variable globale : *tFileList_1.CURRENT_FILEPATH*

- Paramétrez les autres champs comme vous le souhaitez, définissez les séparateurs de lignes et de champs (**Row** et **Field Separator**) en fonction de la structure de votre document.
- Puis définissez manuellement le schéma du fichier à l'aide du bouton **Edit schema** ou sélectionnez le schéma dans le **Repository**. En version Java, assurez-vous que le type de données est correctement défini, conformément à la nature des données traitées.
- Modifiez l'encodage s'il ne correspond pas à celui de votre fichier.
- Dans la zone Output du **tMap**, ajoutez deux tables de sortie, une que vous nommerez *mum* pour la table parent et une deuxième que vous nommerez *baby* pour la table enfant.
- Cliquez sur la colonne *Name* de la zone **Input** et glissez-la dans la table *mum*.
- Cliquez sur la colonne *Years* de la zone **Input** et glissez-la dans la table *baby*.

- Assurez-vous que la table *mum* est bien au dessus de la table *baby* car l'ordre des tables détermine la séquence des flux et ainsi l'insertion de base de données est exécutée correctement.
- Puis sélectionnez une connexion de sortie de type Row pour acheminer correctement le flux vers le composant Database de sortie correspondant.
- Dans l'onglet **Basic settings** des deux composants **tMysqlOutput**, cochez la case **Use an existing connection** pour récupérer les informations du composant **tMysqlConnection**.

- En version Perl, le champ **Commit every** n'apparaît plus puisque vous utilisez le composant **tMysqlCommit** au lieu de gérer les commits de transaction globale. En version Java, ignorez ce champ puisque cette commande sera annulée par le composant **tMysqlCommit**.

- Nommez votre table dans le champ **Table** en vous assurant que vous saisissez la bonne table. Dans cet exemple, la table est soit *f1090_mum*, soit *f1090_baby*.
- Ne définissez aucune action dans le champ **Action on table** puisque les tables ont déjà été créées.
- Sélectionnez **Insert** dans le champ **Action on data** pour les deux composants de sortie (**tMysqlOutput**).
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma défini dans le **tMap**.
- Modifiez le type d'encodage si nécessaire.
- Dans la zone **Additional columns** de l'onglet **Advanced settings** du composant tMysqlOutput correspondant à la table enfant (*f1090_baby*), définissez la colonne *id_baby* de manière à ce que l'*id* de la table parent soit réutilisé.
- Dans le champ **SQL expression**, saisissez : *'(Select Last_Insert_id())'*
- Dans la colonne **Position**, sélectionnez l'option *Before* et dans **Reference column**, l'option *years*.
- Ajoutez le composant **tMysqlCommit** à votre Job et connectez-le au composant **tFileList** à l'aide d'une connexion de type **OnComponentOk** afin que votre Job se termine avec un commit de transaction.
- Dans la vue **Component** du **tMysqlCommit**, sélectionnez dans la liste la connexion à utiliser.

Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
mysql> select * from f1090_mum
    -> ;
+----+--------+
| id | names  |
+----+--------+
|  6 | john   |
|  7 | bruce  |
|  8 | beth   |
|  9 | andrew |
| 10 | donald |
| 11 | betty  |
| 12 | john   |
| 13 | bruce  |
| 14 | beth   |
| 15 | andrew |
| 16 | donald |
| 17 | betty  |
+----+--------+
12 rows in set (0.00 sec)
```

```
mysql> select * from f1090_baby
    -> ;
+---------+-------+
| id_baby | years |
+---------+-------+
|       6 |    10 |
|       7 |    23 |
|       8 |    34 |
|       9 |    10 |
|      10 |    23 |
|      11 |    34 |
|      12 |    10 |
|      13 |    23 |
|      14 |    34 |
|      15 |    10 |
|      16 |    23 |
|      17 |    34 |
+---------+-------+
12 rows in set (0.00 sec)
```

L'*id* de la table parent a été réutilisé pour alimenter la colonne *id_baby*.

tMysqlInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tMysqlInput

Famille de composant	Databases/MySQL	
Fonction	Le composant **tMysqlInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tMysqlInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tMysqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de **Talend Integration Suite Studio**.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant (**Repository**). Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à lire.

	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**. Lorsque vous devez traiter des données au format date/heure *0000-00-00 00:00:00* utilisant ce composant, définissez les paramètres comme suit : `noDatetimeStringSync=true&zeroDateTimeBehavior=convertToNull`.
	Enable stream	Cochez cette case pour déterminer avec quelles lignes vous souhaitez travailler. Cette option permet d'améliorer les performances.
	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données MySQL.	

Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie

Dans ce scénario, vous allez lire des colonnes dynamiques d'une base de données MySQL, les mapper puis les écrire dans une table, dans un fichier de sortie local. En définissant une colonne dynamique et les noms connus des colonnes, vous pouvez retrouver toutes les colonnes de la table de la base de données, y compris les colonnes inconnues.

- Déposez un composant **tMysqlInput**, un composant **tMap** et un composant **tFileOutputDelimited** dans l'espace de modélisation graphique.

- Reliez le **tMysqlInput** au **tMap** à l'aide d'un lien **Row** > **Main**.

- Reliez le **tMap** au **tFileOutputDelimited** à l'aide d'un lien **Row** > ***New Output* (Main)**.
- Double-cliquez sur le **tMysqlInput** afin d'ouvrir sa vue **Basic Settings** dans l'onglet **Component**.

La fonction de schéma dynamique n'est supportée qu'en mode **Built-In**.

- Dans la liste **Property Type**, sélectionnez **Built-in**.
- Dans la liste **DB Version**, sélectionnez la version de la base de données que vous utilisez.
- Dans le champ **Host**, saisissez l'adresse IP du serveur de la base de données.
- Dans le champ **Port**, saisissez le numéro du port d'écoute du serveur de la base de données.
- Dans les champs **Username** et **Password**, saisissez respectivement votre identifiant et votre mot de passe de connexion au serveur de la base de données.
- Dans la liste **Schema type**, sélectionnez **Built-in** puis cliquez sur **Edit schema** afin de définir le schéma dynamique.
  L'éditeur du schéma s'ouvre :

- Cliquez sur le bouton pour ajouter une ligne au schéma.
- Dans les colonnes **Column** et **Db Column**, cliquez dans les champs afin de saisir le nom des colonnes correspondantes.
- Cliquez sur la liste **Type** pour définir le type des données.
  Cliquez sur la flèche et sélectionnez **Dynamic** dans la liste.

Dans la liste **Type**, *le type de la colonne dynamique doit être défini comme* **Dynamic**.

- Cliquez sur **OK** pour fermer l'éditeur du schéma.
- Dans le champ **Table Name**, cliquez sur le bouton **[...]** afin de sélectionner la table souhaitée de la base de données.
  Une boîte de dialogue affiche une arborescence de toutes les tables de la base de données sélectionnée :

- Cliquez sur la table souhaitée, puis cliquez sur **OK** pour fermer la boîte de dialogue.
- Définissez le type de requête, dans la liste **Query Type**, comme **Built-In**.
- Dans le champ **Query**, saisissez la requête permettant de récupérer toutes les colonnes de la table.

*Dans l'instruction SELECT, il est nécessaire d'utiliser le caractère *, pour récupérer toutes les colonnes de la table sélectionnée.*

- Cliquez sur le **tMap** afin d'ouvrir sa vue **Basic Settings** dans l'onglet **Component**.
- Cliquez sur le bouton **[...]** à côté du champ **Map Editor** pour mapper la colonne du fichier source.

- Déposez la colonne définie comme dynamique du schéma d'entrée, à gauche, dans le schéma de sortie, à droite.
  La colonne déposée dans le schéma de sortie contient ses valeurs d'origine.

La colonne dynamique doit être mappée seule et ne peut subir de transformation. Elle ne peut être utilisée dans une expression de filtre ou dans des variables. Elle ne peut être renommée dans la table de sortie et ne peut être utilisée comme condition de jointure.

- Cliquez sur **OK** pour fermer l'éditeur **Map Editor**.
- Double-cliquez sur le composant **tFileOutputDelimited** afin de définir ses **Basic Settings** dans l'onglet **Component**.

- A côté du champ **File Name**, cliquez sur le bouton **[...]** afin de parcourir votre répertoire jusqu'à l'endroit où vous souhaitez sauvegarder votre fichier de sortie, puis saisissez un nom pour le fichier.
- Cochez la case **Include Header** pour récupérer le nom des colonnes ainsi que les données.
- Sauvegardez votre Job et appuyez sur **F6** pour l'exécuter.
  Le fichier de sortie contient le nom de toutes les colonnes et les données correspondantes, récupérés de la base de données via le schéma dynamique :

id	firstname	lastname	dateofbirth	age
1	CALVIN	Cleveland	19/05/1991	60
2	ULYSSES	Harrison	01/11/1986	20
3	CHESTER	Van Buren	26/12/1994	34
4	GEORGE	Reagan	05/10/1988	96
5	JAMES	Grant	17/04/1974	67
6	WILLIAM	Arthur	23/10/1958	81
7	GEORGE	Van Buren	28/02/1954	37
8	ULYSSES	Harrison	20/07/1974	81
9	JIMMY	Fillmore	22/03/1997	20
10	ABRAHAM	Cleveland	31/10/1991	66
11	THOMAS	Harrison	19/10/1975	96
12	MARTIN	Fillmore	11/06/1972	99
13	RONALD	Eisenhower	25/08/1972	26
14	THEODORE	Jefferson	29/09/1997	14
15	ANDREW	Harding	18/09/1954	5

Le Job peut également être exécuté en mode **Traces Debug**, qui vous permet de voir les lignes telles qu'elles sont écrites dans le fichier de sortie, dans l'espace de modélisation graphique.

Pour plus d'informations concernant la définition et le mapping des schémas dynamiques, consultez la section *Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.

Pour un exemple d'écriture de colonnes dynamiques dans une base de données, consultez *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507*.

Pour des scénarios associés, consultez :

- *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190*.
- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239*.
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241*.
- *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507*.

tMysqlLastInsertId

Propriétés du tMysqlLastInsertId

Famille de composant	Databases/MySQL	
Fonction	**tMysqlLastInsertId** affiche les derniers ID ajoutés à une table à partir d'une connexion MySQL spécifiée.	
Objectif	**tMysqlLastInsertId** permet de récupérer les clés primaires des entrées dernièrement ajoutées par un utilisateur à une table MySQL.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Component list	Sélectionnez le composant **tMysqlConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est un composant intermédiaire. *Si vous utilisez ce composant avec un* **tMysqlOutput**, *vérifiez que la case* **Extend Insert** *de l'onglet* **Advanced settings** *est bien décochée. En effet,* **Extend Insert** *permet de faire un chargement par lot, ainsi seul l'ID de la dernière ligne du dernier lot sera retourné.*	
Limitation	n/a	

Scénario : Récupérer les ID des dernières entrées ajoutées

Le scénario Java suivant permet d'ouvrir une connexion à une base de données MySQL, d'y écrire des données spécifiées et enfin de récupérer les derniers ID ajoutés à la connexion existante.

- Cliquez sur chacun des composants suivants et déposez-les dans l'éditeur graphique : **tMySqlConnection**, **tMySqlCommit**, **tFileInputDelimited**, **tMySqlOutput**, **tMysqlLastInsertId**, et **tLogRow**.
- Reliez le **tMySqlConnection** au **tFileInputDelimited** à l'aide d'un lien **OnSubJobOk**.
- Reliez le **tFileInputDelimited** au **tMySqlCommit** à l'aide d'un lien **OnSubJobOk**.

- Reliez les trois derniers composants au **tFileInputDelimited** et entre eux à l'aide de liens **Row Main**, comme suit :

- Sélectionnez le **tMysqlConnection** en cliquant dessus à partir de l'éditeur graphique.
- Cliquez sur la vue **Component** pour en définir la configuration de base (Basic settings).
- Dans l'onglet **Basic settings**, entrez les paramètres de connexion manuellement ou sélectionnez-les à partir d'une liste de variables de contexte, si vous les avez stockées localement sous le noeud **DB connections** de la partie **Metadata** (Métadonnées) du **Repository**. Vous accéderez à cette liste en cliquant dans le champ à renseigner puis en appuyant sur **Ctrl+Espace**. Pour plus d'information sur les Métadonnées, consultez le chapitre *Centraliser les métadonnées* du **Guide Utilisateur** de **Talend Open Studio**.

- Sélectionnez le **tMysqlCommit** en cliquant dessus à partir de l'éditeur graphique, puis cliquez sur la vue **Component** pour en définir la configuration de base (Basic settings).
- Dans le champ **Component list**, sélectionnez le composant **tMysqlConnection** approprié si vous effectuez un Job à plusieurs connexions.
- Sélectionnez le **tFileInputDelimited** en cliquant dessus à partir de l'éditeur graphique.

- Cliquez sur la vue **Component** pour en définir la configuration de base (Basic settings).

- Dans la liste **Property Type**, sélectionnez l'option **Built-In**.
- Renseignez le chemin d'accès au fichier à traiter dans le champ **File Name**. Dans cet exemple, le nom du fichier est *Customers*.
- Dans le champ **Row separator**, définissez le séparateur de lignes permettant d'identifier les fins de ligne. De même, pour **Field separator**, définissez le séparateur de champs permettant de délimiter les différents champs pour chaque ligne.
- Définissez, si nécessaire, le nombre de lignes d'en-tête et de pied-de-page à ignorer, ainsi que le nombre maximum de lignes que vous souhaitez traiter. Dans cet exemple, le fichier comporte une ligne d'en-tête.
- Dans la liste **Schema**, sélectionnez l'option **Built-In**, puis cliquez sur les trois points **[...]** à côté de la mention **Edit Schema** pour définir les données à passer au composant suivant.

Sujets associés : consultez les chapitres *Paramétrer un schéma built-in* et *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.

Dans ce scénario, le schéma se compose de deux colonnes, *name* et *age*. La première colonne comprend le nom des employés et la seconde comporte leurs âges respectifs.

- Sélectionnez le **tMysqlOutput** en cliquant dessus à partir de l'éditeur graphique.
- Cliquez sur la vue **Component** pour en définir la configuration de base (**Basic settings**).

- Cochez la case **Use an existing connection**.
- Dans le champ **Table**, renseignez le nom de la table dans laquelle vous voulez écrire la liste des employés. Dans cet exemple, il s'agit de la table *employee*.
- Sélectionnez les actions à effectuer dans les champs **Action on table** et **Action on data**, à partir des listes proposées. Dans cet exemple, on ne veut effectuer aucune action sur la table elle-même mais exécuter une action *Insert* sur les données.
- Dans la liste **Schema**, sélectionnez l'option **Built-In** puis cliquez sur le bouton **Sync columns** pour synchroniser les colonnes par rapport au composant précédent. Dans cet exemple, le schéma à reproduire dans la table de la base de données MySQL se compose des deux colonnes *name* et *age*.

- Sélectionnez le **tMySqlLastInsertId** en cliquant dessus à partir de l'éditeur graphique.
- Cliquez sur la vue **Component** pour en définir la configuration de base (Basic settings).

- Dans le champ **Component list**, sélectionnez le composant **tMysqlConnection** approprié si vous effectuez un Job à plusieurs connexions.
- Dans la liste **Schema**, sélectionnez l'option **Built-In** puis cliquez sur le bouton **Sync columns** pour synchroniser les colonnes par rapport au composant précédent. Dans le schéma de sortie du **tMySqlLastInsertId**, vous pouvez voir la colonne last_insert_id qui indique les dernières ID ajoutées à la connexion existante.

- Sélectionnez le **tLogRow** en cliquant dessus à partir de l'éditeur graphique, puis cliquez sur la vue **Component** pour en définir la configuration de base (Basic settings). Pour plus d'informations, consultez le *tLogRow, page 1163*.
- Sauvegardez votre Job puis appuyez sur **F6** pour l'exécuter.

```
Starting job lastinserted at 11:12 01/09/2008.
.--------+---+--------------.
|         tLogRow_1          |
|=-------+---+--------------=|
|name    |age|last_insert_id|
|=-------+---+--------------=|
|Marie   |24 |40            |
|Patrick |22 |41            |
|Pierrick|27 |42            |
'--------+---+--------------'

Job lastinserted ended at 11:12 01/09/2008. [exit code=0]
```

Le composant **tMysqlLastInsertId** affiche les dernières ID ajoutées pour chaque ligne de la connexion existante.

tMysqlOutput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tMysqlOutput

Famille de composant	Databases/MySQL	
Fonction	Le composant **tMysqlOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tMysqlOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	DB Version	Sélectionnez la version de MySQL que vous utilisez.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tMysqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de **Talend Integration Suite Studio**.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. **Insert or update on duplicate key or unique index** : Ajouter des entrées si la valeur insérée n'existe pas ou mettre à jour les entrées si la valeur insérée existe déjà et qu'un risque de violation d'une clé unique se présente. **Insert ignore** : Ajouter uniquement de nouvelles lignes afin d'empêcher les erreurs de doublons de clés. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).* La fonction de schéma dynamique peut être utilisée dans différents modes : **Insert**, **Update**, **Insert or Update**, **Update or insert**, **Delete**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.

		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Extend Insert	Cochez cette case pour insérer un ensemble de lignes définies au lieu d'insérer les lignes une par une. Cette option permet un important gain de performance. **Number of rows per insert** : saisissez le nombre de lignes à insérer en un bloc. Notez que si vous sélectionnez un nombre important de lignes, cela peut augmenter la quantité de mémoire utilisée et donc diminuer les performances. Cette option n'est pas compatible avec le lien **Reject**. Vous devez donc la décocher si vous utilisez un lien **Reject** en sortie du composant.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.

		Reference column : Saisissez une colonne de référence que le composant **tMysqlOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Use Hint Options	Cochez cette case pour activer la zone de configuration des indicateurs (ou Hints) permettant d'optimiser l'exécution d'une requête. Dans cette zone, les paramètres sont : - **HINT** : spécifiez l'indicateur dont vous avez besoin, en utilisant la syntaxe `/*+ */`. - **POSITION** : spécifiez la place de l'indicateur dans une instruction SQL. - **SQL STMT** : sélectionnez l'instruction SQL que vous souhaitez utiliser.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus d'écriture dans la base de données.
	Use duplicate key update mode insert	Met à jour les valeurs des colonnes spécifiées en cas de doublon de la clé primaire. **Column** : Saisissez entre guillemets le nom de la colonne à mettre à jour. **Value** : Saisissez l'opération que vous souhaitez effectuer sur la colonne. Pour utiliser cette option, vous devez préalablement sélectionner le mode **Insert** dans la liste **Action on data** de la vue **Basic settings**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Ce composant doit être utilisé en tant que composant de sortie. Il permet de faire des actions sur une table ou les données d'une table d'une base de données MySQL. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation du **tMysqlOutput**, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501*.	

Scénario 1 : Ajouter une colonne et modifier les données

Ce scénario Java est un Job à trois composants permettant de créer de manière aléatoire des données de test à l'aide du composant **tRowGenerator**, dupliquer la colonne à modifier à l'aide du composant **tMap** et de modifier les données à insérer en utilisant une expression SQL dans le composant **tMysqlOutput**.

- Cliquez et déposez les composants suivants dans l'éditeur graphique : **tRowGenerator**, **tMap** et **tMysqlOutput**.
- Connectez-les à l'aide de liens **Row Main**.

- Dans l'éditeur graphique, double-cliquez sur le **tRowGenerator** pour afficher l'onglet **Basic settings**.

- Dans la liste **Schema**, sélectionnez l'option **Built-In**.
- Cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour définir les données à transmettre au composant **tMap**, dans ce scénario, les deux colonnes *name* et *random_date*.

- Cliquez sur **OK** pour fermer la boîte de dialogue.
- Cliquez sur le bouton **[...]** à côté du champ **RowGenerator Editor** pour ouvrir l'éditeur du **tRowGenerator** pour paramétrer les données à générer.

- Cliquez dans le champ **Functions** correspondant et sélectionnez une fonction pour chacune des deux colonnes, `getFirstName` pour la colonne *name* et `getrandomDate` pour la colonne *random_date*.
- Dans le champ **Number of Rows for Rowgenerator**, saisissez 10 pour ne générer que dix lignes de prénoms et cliquez sur **Ok** pour fermer l'éditeur.
- Double-cliquez sur le composant **tMap** pour ouvrir l'éditeur du **tMap**. L'éditeur s'ouvre et affiche les données d'entrée du composant **tRowGenerator**.

- Dans le panneau **Schema editor** de l'éditeur du tMap, cliquez sur le bouton **[+]** de la table de sortie pour ajouter deux lignes et nommez la première colonne *random_date* et la deuxième *random_date1*.

Dans ce scénario, l'objectif est de dupliquer la colonne *random_date* et adapter le schéma afin de modifier les données à transférer dans le composant de sortie.

- Dans l'éditeur du **tMap**, glissez la ligne *random_date* de la table d'entrée vers les lignes *random_date* et *random_date1* de la table de sortie.

- Cliquez sur **OK** pour fermer l'éditeur.
- Dans l'éditeur graphique, double-cliquez sur le composant **tMysqlOutput** pour afficher l'onglet **Basic settings** et paramétrer ses propriétés.

- Dans la liste **Property Type**, sélectionnez l'option **Repository** et cliquez sur le bouton **[...]** pour ouvrir la boîte de dialogue **[Repository content]** et sélectionnez la connexion à la base de données adéquate. Les informations de connexion sont renseignées automatiquement.

  Si vous n'avez pas stocké de connexion à votre base de données sous le noeud **DB connections** sous le noeud **Metadata** du **Repository**, sélectionnez l'option **Built-in** dans la liste Property Type et renseignez les informations de connexion manuellement.

- Cliquez sur le bouton **[...]** à côté du champ **Table** et sélectionnez la table à modifier : la table *Dates* dans ce scénario.
- Dans la liste **Action on table**, sélectionnez l'option **Drop table if exists and create** et sélectionnez *Insert* dans la liste **Action on data**.
- Si nécessaire, cliquez sur **Sync columns** pour récupérer le schéma de colonnes du composant **tMap**.
- Cliquez sur l'onglet **Advanced settings** pour afficher la vue correspondante et définir les paramètres avancés du composant.

- Dans la zone **Additional Columns**, paramétrez les modifications à appliquer aux colonnes. Dans ce scénario, la colonne *One_month_later* remplace la colonne *random_date_1* et les données aussi sont modifiées à l'aide d'une requête SQL ajoutant un mois à la date générée aléatoirement dans la colonne *random_date_1*. Par exemple : 2007-08-12 devient 2007-09-12.
  -Saisissez *One_Month_Later* dans la cellule **Name**.
  -Dans la cellule **SQL expression**, saisissez la requête SQL ajoutant un mois, dans ce scénario : `"adddate(Random_date, interval 1 month)"`.
  -Sélectionnez **Replace** dans la liste **Position**.
  -Sélectionnez *Random_date1* dans la liste **Reference column**.

  Pour ce Job, vous dupliquez la colonne *random_date_1* dans la table avant de la remplacer par la colonne *One_Month_Later*. Le but de cette manipulation était de voir les modifications apportées en amont.

- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

La nouvelle colonne *One_month_later* remplace la colonne *random_date1* dans la base de données et ajoute un mois à chaque date générée aléatoirement.

Random_date	One_Month_Later
2007-03-13 21:52:09	2007-04-13 21:52:09
2008-11-02 15:33:29	2008-12-02 15:33:29
2008-05-17 18:47:15	2008-06-17 18:47:15
2007-09-06 17:44:36	2007-10-06 17:44:36
2008-11-17 11:46:36	2008-12-17 11:46:36
2007-05-19 00:46:58	2007-06-19 00:46:58
2008-04-20 15:36:00	2008-05-20 15:36:00
2007-07-24 21:06:08	2007-08-24 21:06:08
2008-04-03 13:24:17	2008-05-03 13:24:17
2007-06-29 13:22:23	2007-07-29 13:22:23

Voir également : le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput**.

Scénario 2 : Effectuer des mises à jour dans une base de données

Ce scénario Java est un Job à deux composants permettant de mettre à jour les données d'une table MySQL à partir d'un fichier délimité.

- A partir de la **Palette**, cliquez et déposez les composants **tFileInputDelimited** et **tMysqlOutput** dans l'éditeur graphique puis connectez-les à l'aide d'un lien **Row Main**.

- Dans l'éditeur graphique, double-cliquez sur le **tFileInputDelimited** pour afficher l'onglet **Basic settings**.

- Dans le champ **Property Type**, sélectionnez le mode **Repository** si vous avez stocké les informations du fichier délimité sous le noeud **Metadata** du **Repository** ou sélectionnez le mode **Built-In** pour les définir manuellement. Dans ce scénario, on utilise le mode **Built-In**.

- Dans le champ **File Name**, renseignez manuellement le chemin d'accès au fichier contenant les mises à jour à propager dans la base de données ou cliquez sur le bouton **[...]** pour parcourir vos dossiers jusqu'à ce fichier. Dans cet exemple, on utilise le fichier *customer_update*, qui comporte quatre colonnes, *id*, *CustomerName*, *CustomerAddress* et *idState*.

id	CustomerName	CustomerAddress	idState
858	Froggy's Gourmet Catering	1831 Beverly Place #9D	4
859	Dependable Plumbing and Sewer	1550 Ridge Rd.	25
860	Lickmen Restoration	1235 Easton Rd.	40
861	Acturial Enterprises Ltd.	3148 cottonwood Ct.	18
862	Rythmics Ltd.	857 Woodbine Rd	30
863	Acturial Enterprises Ltd.	1482 Concorde Circle	48
864	Crosstracks Car Wash	218 Oakridge Ave.	39
865	Meonits & Mogogni Inc.	616 Cobblestone Cir.	17
866	Foy Aviation	2220 Grant Blvd.	50
867	Ebert Music Center	12 Broadview Lane	29
868	janice Mann Accounting Service	1660 Park Ave.	9
869	Johnson, Erico & Co CPA's	2922 Twin Oaks Drive	40
870	Corbins, Rodriguez, & Savocchi	115 Pleasent Ave.	18
871	Nina's Snow Plowing	3385 University Ave.	20
872	Darcy Frame and Matting Servic	1101 Deerfield Place	47
873	Marks, Marks, and Kaplan Ltd.	1949 Cloverdale Rd.	9

- Définissez si nécessaire les séparateurs de lignes (**Row separator**) et de champs (**Field separator**), l'en-tête (**Header**) et le pied-de-page (**Footer**), ainsi que le nombre de lignes à traiter (**Limit**). La première ligne, qui porte le nom des colonnes, est ignorée dans cet exemple, c'est pourquoi le champ **Header** est défini à "1".

- Dans le champ **Schema**, sélectionnez le mode **Built in** puis cliquez sur le bouton **[...]** à côté du champ **Edit Schema** pour définir les données à passer au composant suivant. Dans cet exemple, le schéma est constitué de quatre colonnes, *id*, *CustomerName*, *CustomerAddress* et *idState*.

Column	Key	Type	Nullable
id	☑	Integer	☑
CustomerName	☐	String	☑
CustomerAddress	☐	String	☑
idState	☐	Integer	☑

- Devant le nom de chaque colonne se trouve une case *Key*. Cochez cette case pour la ou les colonne(s) que vous voulez définir comme clé.

  Vous devez obligatoirement définir au moins une clé pour que le Job s'exécute. Dans le cas contraire, le Job s'arrête automatiquement et un message d'erreur s'affiche dans la console de log.

- Dans l'éditeur graphique, double-cliquez sur le composant **tMysqlOutput** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component** :

- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent. Vous pouvez cliquer sur le bouton **[...]** à côté du champ **Edit schema** pour consulter le schéma et vérifier qu'au moins une colonne a été définie comme clé.
- Dans le champ **Property Type**, sélectionnez **Repository** si vous avez stocké les informations de connexion à la base de données sous le noeud **Metadata** du **Repository** ou sélectionnez le mode **Built-In**, puis renseignez manuellement les champs suivants : **Host**, **Port**, **Database**, **Username** et **Password**.
- Dans le champ **Table**, saisissez le nom de la table à mettre à jour.
- Dans le champ **Action on table**, sélectionnez l'opération que vous souhaitez effectuer sur la table. Pour ce scénario, sélectionnez **None** car la table existe déjà.
- Dans le champ **Action on data**, sélectionnez l'opération que vous souhaitez effectuer sur les données. Pour ce scénario, sélectionnez l'option **Update** pour mettre la table à jour.
- Enregistrez le Job puis appuyez sur **F6** pour l'exécuter.

id	CustomerName	CustomerAddress	idState
858	Froggy's Gourmet Catering	1831 Beverly Place #9D	4
859	Dependable Plumbing and Sewer	1550 Ridge Rd.	25
860	Lickmen Restoration		
861	Acturial Enterprises Ltd.		
862	Rythmics Ltd.		
863	Acturial Enterprises Ltd.		
864	Crosstracks Car Wash		
865	Meonits & Mogogni Inc.		
866	Foy Aviation		
867	Ebert Music Center		
868	janice Mann Accounting Servic		
869	Johnson, Erico & Co CPA's		
870	Corbins, Rodriguez, & Savocch		
871	Nina's Snow Plowing		
872	Darcy Frame and Matting Serv		
873	Marks, Marks, and Kaplan Ltd.		

id	CustomerName	CustomerAddress	idState
858	Froggy's Gourmet Catering	1831 Beverly Place #9-11D	4
859	Dependable Plumbing and Sewer	1550 Ridge Rd.	25
860	Lickmen Restoration	1235 Easton Rd.	40
861	Acturial Enterprises Ltd.	3148 Cottonwood Ct.	18
862	Rythmics Ltd.	857 Woodbine Rd	30
863	Acturial Enterprises Ltd.	1482 Concorde Circle	48
864	Crosstracks Car Wash	218 Oakridge Ave.	39
865	Meonits & Mogogni Inc.	616 Cobblestone Cir.	17
866	Foy Aviation	2220 Grant Blvd.	50
867	Ebert Music Center	12 Broadview Lane	29
868	janice Mann Accounting Service	1660 Park Ave.	9
869	Johnson, Erico & Co CPA's	2922 Twin Oaks Drive	40
870	Corbins, Rodriguez, & Savocchi	115 Pleasent Ave.	18
871	Nina's Snow Plowing	3385 University Ave.	20
872	Darcy Frame and Matting Servic	1101 Deerfield Place	47
873	Marks, Kaplan and Jones Ltd.	1949 Cloverdale Rd.	9

A partir de votre explorateur de base de données, vous pouvez vérifier que la table *customers* a bien été mise à jour. Comme avant sa mise à jour, elle se présente sous la forme des quatre colonnes *id*, *CustomerName*, *CustomerAddress* et *idState*, mais certains champs ont été modifiés selon les données du fichier délimité *customer_update*.

Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject

Ce scénario décrit un Job à quatre composants effectuant une migration d'un fichier client vers la table d'une base de données MySQL et redirigeant les données erronées vers un fichier CSV à l'aide d'un lien **Reject**.

- Dans le **Repository**, sélectionnez la métadonnée correspondant au fichier client que vous souhaitez migrer et glissez-la dans l'espace de modélisation. Dans la boîte de dialogue **[Components]**, sélectionnez **tFileInputDelimited** et cliquez sur **OK**. Les propriétés du composant seront automatiquement renseignées.

- Si vous n'avez pas enregistré les informations concernant votre fichier client sous le noeud **Metadata** du **Repository**. Glissez un composant **tFileInputDelimited** de la famille **File** > **Input** de la **Palette** et renseignez ses propriétés manuellement dans la vue **Component**.
- A partir de la **Palette**, glissez un composant **tMap** de la famille **Processing** dans l'espace de modélisation.
- Dans le **Repository**, développez le noeud **Metadata** et **Db Connections,** et sélectionnez la métadonnée de connexion à la base de données dans laquelle vous souhaitez migrer vos données et glissez-la dans l'espace de modélisation. Dans la boîte de dialogue **[Components]**, sélectionnez **tMysqlOutput** et cliquez sur **OK**. Les propriétés de connexion à la base de données seront automatiquement renseignées.
- Si vous n'avez pas enregistré les informations de connexion à votre base de données sous le noeud **Db Connections** du **Repository**. Glissez un composant **tMysqlOutput** de la famille **Databases** de la **Palette** et renseignez ses propriétés manuellement dans la vue **Component**.

Pour plus d'informations, consultez *Paramétrer un schéma built-in* et *Paramétrer un schéma du Repository* du **Guide Utilisateur** **Talend Open Studio**.

- A partir de la **Palette**, sélectionnez un composant **tFileOutputDelimited** de la famille **File** > **Output** et glissez-le dans l'espace de modélisation.
- Reliez les composant **customers** et **tMap**, et les composants **tMap** et **Localhost** à l'aide d'un lien de type **Row Main**. Vous nommerez ce deuxième lien *out*.
- Reliez le composant **Localhost** au **tFileOutputDelimited** à l'aide d'un lien de type **Row** > **Reject**.
- Double-cliquez sur le composant **customers** pour afficher la vue **Component** correspondante.

- Dans la liste **Property Type**, sélectionnez le mode **Repository** puis cliquez sur le bouton **[...]** correspondant pour sélectionner la métadonnée contenant la connexion à votre fichier. Sinon, sélectionnez le mode **Built-in** et paramétrez manuellement les champs suivants.
- Cliquez sur le bouton **[...]** à côté du champ **File Name**, et indiquez le chemin d'accès et le nom du fichier à utiliser.

- Dans les champs **Row** et **Field Separator**, saisissez entre guillemets les séparateurs de lignes et de champs utilisés dans le fichier.
- Dans les champs **Header**, **Footer** et **Limit**, saisissez le nombre de ligne d'en-tête et de pied-de-page à ignorer, ainsi que le nombre de ligne limite à traiter.
- Dans la liste **Schema**, sélectionnez **Repository** et cliquez sur le bouton **[...]** correspondant pour sélectionner le schéma de votre fichier, s'il est stocké sous le noeud **Metadata** du **Repository**. Sinon cliquez sur le bouton **[...]** correspondant au champ **Edit schema** et définissez manuellement le schéma de votre fichier.

Le schéma se présente comme suit :

Column	Key	Type		N..	Date P...	Length	Precision
id		Integer	✓			2	
CustomerName		String	✓			30	
CustomerAddress		String	✓			25	
idState		Integer	✓			2	
id2		Integer	✓			2	
RegTime		String	✓			16	
RegisterTime		String	✓			23	
Sum1		Float	✓			7	2
Sum2		Float	✓			9	4
Sum3		Float	✓			7	2
Sum4		Float	✓			5	3
Sum5		Float	✓			9	4
Sum6		Float	✓			9	2
Sum7		String	✓			12	

- Double-cliquez sur le **tMap** afin d'ouvrir son éditeur.

- Sélectionnez les colonnes *id*, *CustomerName*, *CustomerAddress*, *idSate*, *id2*, *RegTime*, *RegisterTime* de la zone de gauche et glissez-les dans la table **out** de la zone de droite.

- Dans la zone **Schema editor** en bas de l'éditeur du **tMap**, dans le tableau à droite, changez la longueur de la colonne **CustomerName** en *28* afin de créer une erreur. Ainsi, les données dont la longueur est supérieure à 28 créeront des erreurs qui pourront être récupérées grâce au lien **Reject**.
- Cliquez sur **OK**.
- Dans l'espace de modélisation, double-cliquez sur le composant de sortie **Localhost** pour afficher la vue **Component** correspondante.

- Dans la liste **Property Type**, sélectionnez le mode **Repository** puis cliquez sur le bouton **[...]** correspondant pour sélectionner la métadonnée de connexion à la base de données. Les informations de connexion sont renseignées automatiquement. Sinon, sélectionnez le mode **Built-in** et paramétrez ces champs manuellement.
- Dans le champ **Table**, saisissez le nom de la table à créer. Dans ce scénario, vous l'appelez *customers_data*.
- Dans la liste **Action on data**, sélectionnez l'option **Create table**.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent.
- Assurez-vous que la case **Die on error** est bien décochée, afin que le Job s'exécute malgré l'erreur que vous venez de créer.
- Cliquez sur l'onglet **Advanced settings** de la vue **Component** pour paramétrer les paramètres avancés du composant.

- Décochez la case **Extend Insert** permettant d'insérer des lignes par bloc car cette option n'est pas compatible avec le lien **Reject**.
- Double-cliquez sur le composant **tFileOutputDelimited** pour paramétrer ses propriétés dans la vue **Component**.

- Cliquez sur le bouton **[...]** à côté du champ **File Name** pour indiquer le répertoire de destination et le nom du fichier de sortie.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

1	Griffith Paving	talend@apres	7	41	###########	2001-01-17 06:28	Data truncation: Data too long
5	Terrinni & Sol	770 Exmoor F	5	9	###########	1982-04-19 10:26	Data truncation: Data too long
10	Elle Hypnosis	2032 Northbro	1	7	###########	1975-06-10 20:20	Data truncation: Data too long
15	Darcy Frame	1633 McGove	21	28	###########	2001-01-07 20:40	Data truncation: Data too long
19	Salt & Peppe	965 Marion P	44	33	###########	2006-01-24 10:53	Data truncation: Data too long
31	Arthur M. Fei	1220 Heather	37	24	###########	1974-07-24 20:48	Data truncation: Data too long
36	Got It All Elec	629 Kincaid A	22	28	###########	1993-10-22 15:28	Data truncation: Data too long
53	Crossroads C	2267 Andersc	5	14	###########	1977-10-24 17:53	Data truncation: Data too long
54	Jay's Cabinet	2024 St.John	48	48	2006-02-29 0	2001-01-10 07:48	Data truncation: Data too long
57	Lee Stairwork	2644 Princeto	41	46	###########	1992-04-15 09:41	Data truncation: Data too long
58	Lambert & Lo	662 Lyons Ci	45	17	###########	2002-11-22 04:28	Data truncation: Data too long
70	janice Mann A	3181 Barkwo	28	35	###########	1983-04-20 15:20	Data truncation: Data too long
72	Michael Mont	1077 Court A	9	9	###########	2000-10-07 00:20	Data truncation: Data too long
83	Dean Barnett	2922 Twin Oa	18	46	###########	2006-03-23 10:14	Data truncation: Data too long
96	Doerner's Aw	616 Cobblest	26	28	###########	1993-05-03 04:33	Data truncation: Data too long
##	Auto Interior (	844 Spruce S	48	40	###########	2006-04-25 13:14	Data truncation: Data too long
##	First National	465 Stratford	5	41	###########	2003-09-23 01:54	Data truncation: Data too long
##	Terrinni & Sol	1859 Green E	12	44	###########	1986-12-26 05:48	Data truncation: Data too long

Les données en erreur sont envoyées dans le fichier délimité, ainsi que le type d'erreur rencontrée. Ici, vous avez affaire à des données tronquées : **Data truncation**.

Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL

Dans ce scénario, nous allons lire des colonnes dynamiques à partir d'un fichier source, les mapper puis les écrire dans une table d'une base de données **MySQL**. En définissant une colonne dynamique et les noms connus des colonnes, vous pouvez récupérer toutes les colonnes du fichier source, y compris les colonnes inconnues.

- Déposez un composant **tFileInputDelimited**, un composant **tMap** et un composant **tMysqlOutput** dans l'espace de modélisation graphique.

- Reliez le **tFileInputDelimited** au **tMap** à l'aide d'un lien **Row** > **Main**.
- Reliez le **tMap** au **tMysqlOutput** à l'aide d'un lien **Row** > ***New Output* (Main)**.
- Double-cliquez sur le **tFileInputDelimited** afin d'ouvrir sa vue **Basic Settings**, dans l'onglet **Component**.

La fonction de schéma dynamique n'et supportée qu'en mode **Built-In**.

- Dans la liste **Property Type**, sélectionnez **Built-In** dans la liste.
- Cliquez sur le bouton **[...]** à côté du champ **File name/Stream** et parcourez votre répertoire jusqu'au fichier d'entrée.
- Saisissez les caractères que vous souhaitez utiliser comme séparateurs de lignes et de champs respectivement, dans les champs **Row Separator** et **Field Separator**.
- Cliquez sur **Edit Schema** afin de définir le schéma du fichier source.
  La boîte de dialogue **Edit Schema** s'ouvre.

- Ajoutez ou supprimez autant de lignes que nécessaire, à l'aide des boutons et .
- Modifiez l'ordre des colonnes en utilisant les boutons et .
- Dans la colonne **Column**, saisissez le nom de chaque colonne connue dans une ligne différente.
- Dans la dernière ligne, saisissez un nom pour la colonne dynamique.
- Dans la colonne **Type**, cliquez dans chaque champ pour définir le type de données de la colonne correspondante.
  Cliquez sur la flèche afin de sélectionner le bon type de données.

Dans la colonne **Type**, *le type de la colonne dynamique doit être défini comme* **Dynamic**.

La colonne dynamique doit être définie dans la dernière ligne du schéma.

- Cliquez sur **OK** pour fermer la boîte de dialogue quand vous avez terminé de définir le schéma source.
- Cliquez sur le composant **tMap** afin d'ouvrir sa vue **Basic Settings** dans l'onglet **Component**.

- Cliquez sur le bouton **[...]** à côté du champ **Map Editor** pour mapper les colonnes du fichier source.

- Dans la barre d'outils en haut du panneau **Output Panel**, en haut à droite de la fenêtre, cliquez sur le bouton .
  La boîte de dialogue **Add an Output** apparaît.

- A côté de l'option **New output**, saisissez un nom pour le schéma de sortie.
- Cliquez sur **OK** pour fermer la boîte de dialogue.
- A l'aide de **Ctrl+clic**, sélectionnez le nom des colonnes dans le schéma d'entrée à gauche, et déposez-les dans le schéma de sortie.
  Les colonnes déposées contiennent leurs valeurs originales et sont automatiquement mappées une à une.

- Dans le schéma de sortie, cliquez sur la ligne correspondante, dans la colonne **Expression**, si vous souhaitez utiliser l'**Expression Builder** afin de définir des paramètres avancés pour les colonnes correspondantes.
- Cliquez sur le bouton **[...]** qui apparaît afin d'ouvrir l'**Expression Builder** et définir les paramètres, comme nécessaire.

Pour plus d'informations concernant l'utilisation de l'**Expression Builder**, consultez *Créer du code à l'aide de l'Expression Builder, page 265*.

La colonne dynamique doit être mappée seule et ne peut subir de transformation. Elle ne peut être utilisée dans une expression de filtre ou dans des variables. Elle ne peut être renommée dans la table de sortie et ne peut être utilisée comme condition de jointure.

- Cliquez sur **OK** afin de fermer le **Map Editor**.
- Double-cliquez sur le composant **tMysqlOutput** afin de définir ses **Basic Settings**, dans l'onglet **Component**.

- Dans la liste **Property Type**, sélectionnez **Built-in**.
- Dans la liste **DB Version** sélectionnez la version de la base de données que vous utilisez.
- Dans le champ **Host**, saisissez l'adresse IP du serveur de la base de données.
- Dans le champ **Port**, saisissez le numéro du port d'écoute du serveur de la base de données.
- Saisissez respectivement votre identifiant et votre mot de passe de connexion dans les champs **Username** et **Password**.
- A côté du champ **Action on table**, sélectionnez l'action requise.
- A côté du champ **Action on data**, sélectionnez l'action requise.
- Dans la liste **Schema type** définissez le type en **Built-in** et cliquez sur **Edit schema** afin de modifier la schéma si nécessaire.
- Appuyez sur **F6** pour exécuter le Job.
  La table est écrite dans la base de données MySQL avec les données et les noms de colonnes précédemment inconnus :

id	firstname	lastname	dateofbirth	age
1	CALVIN	Cleveland	19-05-1991	60
2	ULYSSES	Harrison	01-11-1986	20
3	CHESTER	Van Buren	26-12-1994	34
4	GEORGE	Reagan	05-10-1988	96
5	JAMES	Grant	17-04-1974	67
6	WILLIAM	Arthur	23-10-1958	81
7	GEORGE	Van Buren	28-02-1954	37
8	ULYSSES	Harrison	20-07-1974	81
9	JIMMY	Fillmore	22-03-1997	20

Le Job peut également être exécuté en mode **Traces Debug**, ce qui vous permet de voir les lignes comme elles sont écrites dans le fichier de sortie, dans l'espace de modélisation.

Pour plus d'informations concernant la définition et le mapping des schémas dynamiques, consultez la section *Onglet Dynamic settings* dans le **Guide Utilisateur Talend Integration Suite Studio**.

Pour un exemple d'écriture de colonnes dynamiques dans un fichier de sortie, consultez *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478*.

tMysqlOutputBulk

Propriétés du tMysqlOutputBulk

Les composants **tMysqlOutputBulk** et **tMysqlBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tMysqlOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le chargement des données dans la base de données.

Famille de composant	Databases/MySQL	
Fonction	Ecrit un fichier composé de colonnes et basé sur le délimiteur défini et sur les standards MySQL.	
Objectif	Prépare le fichier à utiliser comme paramètre dans la requête INSERT servant à alimenter une base de données MySQL.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Row separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Include header	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.

	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
Utilisation	Ce composant est généralement utilisé avec le composant **tMysqlBulkExec**. Ensemble, ils offrent un gain de performance important pour l'alimentation d'une base de données MySQL.	

Scénario : Insérer des données transformées dans une base MySQL

Ce scénario décrit un Job contenant quatre composants dont le but est d'alimenter un base MySQL à partir d'un fichier paramètres contenant des données transformées. Deux étapes sont requises pour ce Job, d'une part pour créer le fichier paramètres qui sera utilisé dans un second temps. La première étape inclut une phase de transformation des données contenues dans le fichier.

- Glissez et déposez les composants suivants : **tRowGenerator**, **tMap**, **tMysqlOutputBulk** et **tMysqlBulkExec** dans l'espace de modélisation.
- Connectez le flux principal à l'aide de connexions **Row > Main**.
- Connectez le composant de début (**tRowGenerator**, dans cet exemple) au composant **tMysqlBulkExec** à l'aide d'une connexion **trigger** de type **OnSubjobOk**.
- Un composant **tRowGenerator** est utilisé pour générer des données de façon aléatoire. Double-cliquez sur le composant **tRowGenerator** pour lancer l'éditeur.
- Définissez le schéma des lignes à générer et la nature des données à générer. Dans cet exemple, le fichier *clients* à créer contient les colonnes suivantes : *ID*, *First Name*, *Last Name*, *Address*, *City* qui sont toutes de type chaîne de caractères (**string**) à l'exception de l'*ID* qui est de type entier (**integer**).

- Certaines informations du schéma n'ont pas nécessairement besoin d'être affichées. Pour les dissimuler, cliquez sur le bouton **Columns** dans la barre d'outils et décochez les colonnes à cacher, par exemple : **Precision** ou **Parameters**.
- Utilisez le bouton **[+]** pour ajouter autant de colonnes que possible à votre schéma.
- Cliquez sur le bouton **Refresh** en haut à droite de l'éditeur pour visualiser un aperçu des lignes générées en sortie dans l'onglet **Preview** en bas de l'éditeur.
- Cliquez sur **OK** pour valider.
- Dans l'espace de modélisation, double-cliquez sur le composant **tMap** pour ouvrir son éditeur et paramétrer la transformation.
- Faites glisser toutes les colonnes de la table d'entrée (**row1**) vers la table de sortie (**clients**).

- Appliquez la transformation sur la colonne *LastName* en ajoutant `.toUpperCase()` à la fin. Cela mettra les noms des clients en majuscule.
- Cliquez sur **OK** pour valider la transformation.
- Dans l'espace de modélisation, double-cliquez sur le composant **tMysqlOutputBulk** pour afficher sa vue **Component** et paramétrer ses propriétés.

- Définissez le chemin d'accès et le nom du fichier à produire dans le champ **File Name**. Si les propriétés du fichier délimité sont conservées dans le **Repository**, sélectionnez l'entrée correspondante dans le champ **Property type** afin de les récupérer. Dans cet exemple, le nom de fichier est *clients.txt*.
- Le schéma est propagé à partir du composant **tMap**, si vous cliquez sur **Yes** lorsque la boîte de dialogue vous demande de confirmer la propagation. Sinon, cliquez sur le bouton **Sync columns** pour récupérer le schéma.
- Dans cet exemple, n'incluez pas les informations d'en-tête, puisque la table doit déjà les contenir.
- L'encodage est celui par défaut dans ce scénario.
- Cliquez sur **OK** pour valider la sortie.
- Double-cliquez sur le composant **tMysqlBulkExec** pour paramétrer la requête INSERT à exécuter.

- Définissez les informations de connexion à la base de données. Nous vous recommandons de conserver ce type d'informations dans le **Repository**, afin de pouvoir les récupérer à tout moment et pour tous vos Jobs.
- Dans le champ **Table**, saisissez le nom de la table à alimenter, ici, *clients*.
- Dans l'onglet **Advanced settings**, saisissez le séparateur de colonnes, dans le champ **Fields terminated by**.
- Vérifiez que l'encodage défini dans le champ **Encoding** correspond à celui de vos données.
- Appuyez sur **F6** pour exécuter le Job.

Resultset 1

ID	Firstname	Lastname	Address	City
1	Theodore	EISENHOWER	Moreno Drive	Concord
2	Rutherford	COOLIDGE	Lindbergh Blvd	Madison
3	William	JEFFERSON	Katella Avenue	Boise
4	Benjamin	BUCHANAN	French Camp Turnpike Road	Juneau
5	Andrew	WILSON	North Preisker Lane	Frankfort
6	Theodore	MCKINLEY	Pacific Hwy S	Augusta
7	William	POLK	Calle Real	Tallahassee
8	Dwight	HAYES	Timberlane Drive	Des Moines
9	Abraham	BUCHANAN	Padre Boulevard	Des Moines
10	Abraham	MADISON	N Kentwood	Olympia

La table de la base *clients* est alimentée avec les données du fichier notamment les données transformées dans la colonne *Last name*.

Pour une simple opération d'Insert ne nécessitant pas de transformation, l'utilisation du composant **tMysqlOutputBulkExec** permet d'économiser une étape dans le processus et ainsi de gagner en performance.

Voir également les *Propriétés du tMysqlOutputBulkExec, page 518*.

tMysqlOutputBulkExec

Propriétés du tMysqlOutputBulkExec

Les composants **tMysqlOutputBulk** et **tMysqlBulkExec** sont généralement utilisés ensemble comme deux parties d'un processus en deux étapes. Dans la première étape, un fichier de sortie est généré. Dans la deuxième étape, ce fichier est utilisé lors de l'opération d'INSERT afin de populer une base de données. Cette exécution en deux étapes est unifiée dans le composant **tMysqlOutputBulkExec**.

Famille de composant	Databases/MySQL	
Fonction	Le **tMysqlOutputBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tMysqlOutputBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données MySQL.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	DB Version	Sélectionnez la version de MySQL que vous utilisez.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectue aucune action sur la table. **Drop and create the table** : supprime la table puis en créer une nouvelle. **Create a table** : crée une table qui n'existe pas encore. **Create table if doesn't exist** : crée la table si nécessaire. **Clear a table** : supprime le contenu de la table.
	Table	Nom de la table à écrire. Une seule table peut être écrite à la fois et cette table doit déjà exister pour que l'opération d'Insert soit autorisée.

	Local File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
	Append	Cochez cette option pour ajouter de nouvelles lignes à la fin du fichier.
	Schema et *Edit schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
Advanced settings	*Additional JDBC Parameters*	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données.
	Row separator	Chaîne (ex : “\n” sous Unix) séparant les lignes.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Text enclosure	Caractères entourant les données.
	Create directory if not exists	Cette case est cochée par défaut. Cette option permet de créer le dossier contenant le fichier de sortie s’il n’existe pas déjà.
	Custom the flush buffer size	Cochez cette case pour personnaliser la taille de la mémoire utilisée pour stocker temporairement les données, et dans le champ **Row number**, saisissez le nombre de lignes après lesquelles la mémoire est à nouveau libérée.
	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert records in table** : Insère de nouveaux enregistrements dans une table. **Replace records in table** : Remplace les enregistrements existants par de nouveaux. **Update records in table** : Modifie les enregistrements existants. **Ignore records in table** : Ignore les enregistrements existants ou insère les nouveaux.
	Records contain NULL value	Cette case est cochée par défaut. Elle permet de prendre les champs de valeur NULL en compte. Si vous la décochez, les valeurs NULL seront remplacées par des valeurs vides.
	Encoding	Sélectionnez l’encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est principalement utilisé lorsqu’aucune transformation particulière n’est requise sur les données à charger dans la base de données.	
Limitation	n/a	

Scénario : Insérer des données dans une base MySQL

Ce scénario est un Job à deux composants qui effectue le même type d'opération d'alimentation que les composants *tMysqlOutputBulk, page 513* et *tMysqlBulkExec, page 462*, mais sans transformation de données.

- Cliquez et déposez les composants suivants : **tRowGenerator** et **tMysqlOutputBulkExec**.
- Connectez les composants via un lien de type **Row** > **Main**.
- Paramétrez le **tRowGenerator** de la même manière que dans le *Scénario : Insérer des données transformées dans une base MySQL, page 514*. Le schéma est constitué de plusieurs colonnes notamment : *ID*, *First Name*, *Last Name*, *Address* et *City*.
- A partir de l'espace de modélisation, double-cliquez sur le composant **tMysqlOutputBulkExec** pour afficher la vue **Component** et paramétrer ses propriétés.

- Paramétrez la connexion à la base de données si nécessaire, en suivant les recommandations du *Scénario : Insérer des données transformées dans une base MySQL, page 514*, concernant la conservation des informations de connexion dans la partie **Metadata** du **Repository**. Pour cela, sélectionnez **Repository** dans le champ **Property Type** et sélectionnez la connexion adéquate dans le champ adjacent. Les champs suivants seront renseignés automatiquement.

Pour plus d'informations, consultez les sections *Paramétrer un schéma built-in* et *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.

- Dans le champ **Action on table**, sélectionnez l'option **None** puisque vous souhaitez insérer les données dans une table déjà existante.
- Dans le champ **Table**, saisissez le nom de la table à alimenter, ici, *clients*.
- Dans le champ **Local filename**, indiquez le chemin d'accès et le nom du fichier contenant les données à charger dans la table, ici, *clients.txt*.

- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent.
- Cliquez sur l'onglet **Advanced settings** pour paramétrer les propriétés avancées du composant.

- Dans la liste **Action on data**, sélectionnez l'option **Insert records in table** pour insérer ces nouvelles données dans la table.
- Appuyez sur **F6** pour exécuter le Job

Le résultat devrait être relativement identique à celui obtenu dans le *Scénario : Insérer des données transformées dans une base MySQL, page 514*. Cependant les données elles-mêmes peuvent différer légèrement puisque les données sont regénérées de façon aléatoire à chaque exécution.

tMysqlRollback

Propriétés du tMysqlRollback

Ce composant est étroitement lié aux composants **tMysqlCommit** et **tMysqlConnection**. Ils sont généralement utilisés ensemble lors d'une transaction.

Famille de composant	Databases/MySQL	
Fonction	Le **tMysqlRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant **tMysqlConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce component est généralement utilisé avec d'autres composants MySQL, notamment les composants **tMysqlConnection** et **tMysqlCommit** components.	
Limitation	n/a	

Scénario : Annuler l'insertion de données dans des tables mère/fille

En vous basant sur le *Scénario : Insérer des données dans des tables mère/fille, page 471* du composant **tMysqlConnection**, insérez une opération de rollback (annulation) afin d'éviter un commit inutile.

- Cliquez et déposez un composant **tMysqlRollback** et connectez-le au composant de début.

- Paramétrez l'unique champ du Rollback avec la connexion à la base de données adéquate.

Ce nouvel élément ajouté au Job assure que la transaction ne sera pas commitée partiellement.

tMysqlRow

Propriétés du tMysqlRow

Famille de composant	Databases/MySQL	
Fonction	**tMysqlRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tMysqlRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	DB Version	Sélectionnez la version de MySQL que vous utilisez.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tMySQLConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.

	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à traiter.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Guess Query	Cliquez sur le bouton **Guess Query** pour générer la requête correspondant au schéma de votre table dans le champ **Query**.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Propagate QUERY's recordset	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.

	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des “?” dans l’instruction SQL définie dans le champ **Query** de l’onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l’instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l’écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d’exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénario 1 : Supprimer et re-générer un index de table MySQL

Le scénario suivant crée un Job à quatre composants permettant de supprimer un index de table, effectuer un Insert dans cette table avant de re-générer l’index.

- Sélectionnez les composants suivants dans la Palette et déposez-le dans l’espace de modélisation : **tMysqlRow** (x2), **tRowGenerator**, **tMysqlOutput**.
- Connectez le composant **tMysqlIntput** au **tRowGenerator**.
- Puis à l’aide d’une connexion **OnComponentOk**, connectez le premier composant **tMysqlRow** au **tMysqlInput**.

- Connectez le composant **tRowGenerator** au second composant **tMysqlRow** à l'aide d'une connexion **OnSubjobOk**.
- Sélectionnez le composant **tMysqlRow** et renseignez les propriétés de la base de données dans l'onglet **Basic settings**.
- Dans les champs **Property type** et **Schema**, sélectionnez la base de données appropriée dans la liste.
- Les détails de connexion à la base de données ainsi que le schéma de la table sont renseignés automatiquement.
- Propagez les informations de propriétés et de schéma aux autres composants du Job.
- La requête étant conservée dans les **Metadata** du **Repository**, vous pouvez également sélectionner **Repository** dans le champ **Query type** et cliquer sur la requête correspondante.
- Si vous ne conservez pas vos requêtes dans le **Repository**, saisissez la déclaration SQL suivante : *drop index <index_name> on <table_name>*
- Puis sélectionnez le deuxième composant **tMysqlRow**, vérifiez les propriétés et le schéma de base de données.
- Puis saisissez la déclaration SQL permettant de recréer un index à l'aide de la formulation suivante : *create index <index_name> on <table_name> (<column_name>);*
- Le composant **tRowGenerator** permet de générer automatiquement les colonnes à ajouter à la table de sortie définie.
- Sélectionnez le composant **tMysqlOutput** et renseignez les propriétés de connexion à la base de données soit à partir du **Repository** ou manuellement si les informations de connexion ne sont utilisées que pour ce Job. La table à alimenter a pour nom : *comprehensive*.
- Le schéma est automatiquement hérité du flux de données du **tLogRow**. Editez le schéma et vérifiez que sa structure correspond au schéma attendu par la table de base de données spécifiée.
- Dans le champ **Action on table**, sélectionnez *None* et dans le champ **Action on data**, sélectionnez *Insert*.
- Aucune colonne supplémentaire n'est requise pour ce Job.
- Appuyez sur **F6** pour exécuter ce Job.

Si vous avez la possibilité d'observer l'action sur la base de données, remarquez que l'index a été supprimé en début de Job puis recréé à la fin de l'action d'Insert.

Voir également : les propriétés du *tDBSQLRow, page 248*.

Scénario 2 : Utiliser l'instance PreparedStatement pour faire une requête sur des données

Le scénario suivant décrit un Job à quatre composants permettant de mettre en correspondance la colonne d'une table avec un fichier clients. En effet, la table MySQL contient la liste de tous les Etats américains avec leur identifiant, et le fichier contient des informations sur vos clients avec l'identifiant de l'état dans lequel ils résident. L'objectif de ce scénario est de récupérer le nom de l'Etat pour chaque client à l'aide d'une requête SQL. Afin de traiter un grand nombre de données

plus rapidement, utilisez l'instance PreparedStatement permettant de n'exécuter qu'une seule fois la requête au lieu de l'effectuer à chaque ligne, puis chaque ligne de données est envoyé en paramètre.

Pour ce scénario, utilisez un fichier et une base de données dont vous avez préalablement stocké la connexion et les propriétés dans des métadonnées dans le **Repository**. Pour plus d'informations concernant la création de métadonnées de fichiers délimités, consultez la section *Configurer un schéma File Delimited*, pour plus d'informations concernant la création de métadonnées de connexion à une base de données, consultez la section *Configurer une connexion DB*, et pour plus d'informations sur l'utilisation de ces métadonnées, consultez la section *Paramétrer un schéma du Repository* du **Guide de Référence des composants Talend Open Studio**.

- A partir du **Repository**, développez le noeud **Metadata** et **File delimited**
- Sélectionnez la métadonnée correspondant au fichier clients que vous souhaitez utiliser dans ce Job. Ici, vous utilisez la métadonnée *customers*.
- Glissez cette métadonnée dans l'espace de modélisation et double-cliquez sur **tFileInputDelimited** dans la boîte de dialogue **Components** afin de créer un composant **tFileInputDelimited** déjà paramétré.

- Dans la liste **Schema**, sélectionnez **Built-in** afin de modifier le schéma de votre composant. Puis, cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour ajouter la colonne dans laquelle insérer le nom de l'Etat.

Column	Key	Type	N..	Date Patt...	Length	Precision
id		Integer			2	
CustomerName		String			30	
CustomerAddress		String			25	
idState		Integer			2	
id2		Integer			2	
RegTime		String			16	
RegisterTime		String			23	
Sum1		Float			7	2
Sum2		Float			9	4
Sum3		Float			7	2
Sum4		Float			5	3
Sum5		Float			9	4
Sum6		Float			9	2
Sum7		String			12	
LabelStateRecordSet		Object				

- Cliquez sur le bouton **[+]** pour ajouter une colonne au schéma. Renommez cette colonne *LabelStateRecordSet*, et elle est de type objet, donc sélectionnez **Object** dans la liste **Type**. Cliquez sur **OK** pour enregistrer vos modifications.
- A partir de la **Palette**, sélectionnez les composants **tMysqlRow**, **tParseRecordSet** et **tFileOutputDelimited** et glissez-les dans l'espace de modélisation.
- Reliez les quatre composants via des liens de type **Row** > **Main**.
- Double-cliquez sur le composant **tMysqlRow** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component**.

- Dans la liste **Property Type**, sélectionnez **Repository** et cliquez sur le bouton [...] pour sélectionner la connexion à la base de données centralisée dans les métadonnées du Repository. Les champs **DB Version**, **Host**, **Port**, **Database**, **Username** et **Password** sont renseignés automatiquement. Si vous êtes en mode **Built-in**, renseignez ces champs manuellement.
- Dans la liste **Schema**, sélectionnez **Built-in** pour paramétrer le schéma manuellement et ajouter la colonne *LabelStateRecordSet*, ou cliquez directement sur le bouton **Sync columns** pour récupérer le schéma du composant précédent.
- Dans le champ **Query**, saisissez la requête SQL à utiliser. Ici, vous souhaitez récupérer les noms des Etats américains contenus dans la colonne *LabelState* de la table MySQL *us_state* : `"SELECT LabelState FROM us_state WHERE idState=?"`. Le point d'interrogation "?" représente le paramètre à définir dans l'onglet **Advanced settings**.
- Cliquez sur l'onglet **Advanced settings** pour paramétrer les propriétés avancées du composant.

- Cochez la case **Propagate QUERY's recordset** et sélectionnez la colonne *LabelStateRecordSet* dans la liste **use column** pour insérer le résultat de la requête dans cette colonne.
- Cochez la case **Use PreparedStatement** et définissez le paramètre utilisé dans la requête dans le tableau **Set PreparedStatement Parameters**.
- Cliquez sur le bouton **[+]** pour ajouter un paramètre.
- Dans la cellule **Parameter Index**, saisissez la position du paramètre dans l'instruction SQL. Saisissez "*1*" étant donné que vous n'utilisez qu'un seul paramètre dans l'exemple.
- Dans la cellule **Parameter Type**, saisissez le type du paramètre. Ici, le paramètre est de type entier, donc sélectionnez **Int** dans la liste.
- Dans la cellule **Parameter Value**, saisissez la valeur du paramètre. Ici, vous souhaitez récupérer le nom de l'Etat en fonction de leur ID pour chaque client du fichier d'entrée, saisissez donc "`row1.idState`".
- Double-cliquez sur le composant **tParseRecordSet** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component**.

- Dans la liste **Prev. Comp. Column list**, sélectionnez la colonne du composant précédent à analyser. Dans cet exemple, sélectionnez la colonne *LabelStateRecordSet*.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent et le tableau **Attribute table** est automatiquement renseigné avec les colonnes du schéma.
- Dans le tableau **Attribute table**, dans le champ **Value** correspondant à la colonne *LabelStateRecordSet*, saisissez entre guillemets le nom de la colonne contenant le nom des Etats à récupérer et à mettre en correspondance avec chaque client. Dans cet exemple, saisissez "*LabelState*".
- Double-cliquez sur le composant **tFileOutputDelimited** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component**.

- Dans le champ **File Name**, renseignez le chemin d'accès et le nom du fichier de sortie.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

customers.csv

	A	B	C	D	E	F	G	H	O
1	1	Griffith Paving	talend@apres	7	41	03/11/2006 09:20	2001-01-17 06:	67852.0	Connecticut
2	2	Bill's Dive Sh	511 Maple Av	35	5	19/11/2004 15:48	2002-06-07 09:	88792.0	Ohio
3	3	Childress Chi	662 Lyons Ci	1	28	16/02/2005 08:27	1990-04-01 21:	35340.0	Alabama
4	4	Facelift Kitch	220 Vine Ave	41	15	22/08/2002 09:55	1972-04-23 18:	6097.0	South Dakota
5	5	Terrinni & Sor	770 Exmoor F	5	9	28/06/2001 09:15	1982-04-19 10:	5146.0	California
6	6	Kermit the Pe	1860 Parksid	28	15	17/08/2003 10:07	2006-05-27 17:	16087.0	Nevada
7	7	Tub's Furnitur	807 Old Trail	15	9	27/08/2000 03:13	1970-03-27 23:	53216.0	Iowa
8	8	Toggle & Mye	618 Sheriden	9	15	24/03/2006 23:07	2005-08-02 01:	74168.0	Florida
9	9	Childress Chi	788 Tennysor	12	33	10/09/2001 06:33	1994-05-03 11:	92176.0	Idaho
10	10	Elle Hypnosis	2032 Northbr	1	7	11/01/1977 03:07	1975-06-10 20:	48498.0	Alabama

Une colonne contenant le nom de l'Etat américain correspondant à chaque client a été ajoutée au fichier.

tMysqlSCD

Le composant **tMysqlSCD** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tMysqlSCD, page 19*.

tMysqlSCDELT

Le composant **tMysqlSCDELT** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tMysqlSCDELT, page 32*.

tMysqlSP

Propriétés du tMysqlSP

Familles de composant	Databases/Mysql	
Fonction	Le composant **tMysqlSP** appelle une procédure stockée de base de données.	
Objectif	Le **tMysqlSP** permet de centraliser des requêtes multiples ou complexes dans une base de données et de les appeler plus facilement.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier des propriétés du composant. Les champs suivants sont alors pré-remplis.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tMysqlConnection**.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification sur l'utilisateur de base de données.
	Schema et *Edit Schema*	Dans une procédure stockée, le schéma est un paramètre d'entrée. Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le **Repository**. Ainsi, il est réutilisable. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	SP Name	Saisissez le nom exact de la Procédure Stockée (SP).
	Is Function / Return result in	Cochez cette case, si une seule valeur doit être retournée. Sélectionnez dans la liste la colonne du schéma sur laquelle est basée la valeur à obtenir.

	Parameters	Cliquez sur le bouton (+) et sélectionnez dans le champ **Schema Columns** les différentes colonnes nécessaires à la procédure. Notez que le schéma de la SP peut contenir plus de colonnes qu'il n'y a de paramètres utilisés dans la procédure. Sélectionnez le **Type** de paramètre : **IN** : paramètre d'entrée (Input) **OUT** : paramètre de sortie (Output)/valeur retournée **IN OUT** : les paramètres d'entrée doivent être retournées sous forme de valeur, même après modifications via la procédure (fonction). **RECORDSET** : les paramètres d'entrée doivent être retournées sous forme d'ensemble de valeurs, au lieu d'une valeur unique. Consultez le composant *tParseRecordSet* si vous voulez analyser un ensemble d'enregistrements d'une table de données ou d'une requête SQL.
Utilisation	Ce composant est un composant intermédiaire. Il peut être utilisé comme composant de début. Dans ce cas, seuls les paramètres d'entrée sont autorisés.	
Limitation	La syntaxe de la Procédure Stockée doit correspondre à celle de la base de données.	

Scénario : Trouver le libellé State à l'aide d'une procédure stockée

Le Job suivant a pour but de trouver les libellés State en fonction des ID impairs de la colonne State d'une table MySQL à deux colonnes à l'aide d'une procédure stockée.

- Cliquez et déposez les composants suivants dans l'espace de modélisation : **tRowGenerator**, **tMysqlSP** et **tLogRow**.
- Connectez les composants à l'aide d'une connexion de type **Row Main**.
- Le **tRowGenerator** est utilisé pour générer des ID impairs. Double-cliquez sur le composant pour lancer l'éditeur.

- Cliquez sur le bouton **[+]** pour ajouter une colonne au schéma à générer.

- Cochez la case **Key** pour définir l'entrée comme étant une clé primaire et définissez le **Type** en **Int** (entier).
- Dans le champ **Length**, la longueur est de 2 chiffres au maximum.
- Utilisez la fonction préétablie appelée **sequence** mais modifiez les paramètres dans la partie inférieure de la fenêtre.

- Modifiez la valeur (**Value**) de **step** : de *1* à *2* pour cet exemple. Mais la valeur de départ (start value) reste *1*.
- Dans le champ **Number of generated rows**, définissez le nombre de lignes à générer à *25*, afin que tous les ID impairs de State (il y a 50 Etats) soient générés.
- Cliquez sur **OK** pour valider la configuration.
- Puis sélectionnez le composant **tMysqlSP** et paramétrez ses propriétés.

- Dans le champ **Property type**, sélectionnez l'option **Repository** puis sélectionnez l'entrée correspondante dans la liste. Les informations de connexion sont remplies automatiquement.
- Sinon, paramétrez les informations de connexion manuellement.
- Cliquez sur **Sync Column** pour récupérer le schéma généré du composant précédent.
- Puis cliquez sur **Edit Schema** et ajoutez une colonne supplémentaire contenant les libellés des Etats (State) à obtenir en sortie, en plus de l'ID.
- Sélectionnez le type d'encodage dans la liste.
- Dans le champ **SP Name**, saisissez le nom de la procédure comme il a été défini dans la base de données (ici, *getstate*). La procédure à exécuter correspond à :

```
DROP PROCEDURE IF EXISTS `talend`.`getstate` $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `getstate`(IN pid
INT, OUT pstate VARCHAR(50))
BEGIN
 SELECT LabelState INTO pstate FROM us_states WHERE idState =
pid;
END $$
```

- Dans la zone **Parameters**, cliquez sur le bouton (+) pour ajouter une ligne à la table.
- Définissez le champ **Column** en *ID*, et le champ **Type** en *IN*, ainsi il sera considéré comme paramètre d'entrée dans la procédure.
- Ajoutez une deuxième ligne et définissez le champ **Column** en *State* et le champ **Type** en *Out*, ainsi il sera considéré comme le paramètre de sortie a retournée.
- Et enfin, paramétrez les propriétés du composant **tLogRow**.

- Synchronisez le schéma avec celui du composant précédent en cliquant sur le bouton **Sync columns**.
- Et cochez la case **Print values in cells of a table** pour un meilleur confort de lecture.
- Puis enregistrez votre Job et exécutez-le.

```
Starting job MysqlSP at 17:24 23/08/2007.
.--+--------------.
|     tLogRow_1    |
|=-+--------------=|
|ID|State          |
|=-+--------------=|
|1 |Alabama        |
|3 |Arizona        |
|5 |California     |
|7 |Connecticut    |
|9 |Florida        |
|11|Hawaii         |
|13|Illinois       |
|15|Iowa           |
```

La sortie affiche les différents Etats (State) avec leur ID impair comme défini dans la procédure.

Consultez également le composant *tParseRecordSet*, *page 635* si vous voulez analyser un ensemble d'enregistrements d'une table de données ou d'une requête SQL.

tMysqlTableList

Propriétés du tMysqlTableList

Famille de composant	Databases/MySQL	
Fonction	Le composant **tMysqlTableList** effectue une opération d'itération sur toutes les tables d'une base de données, grâce à une connexion MySQL définie.	
Objectif	Le composant **tMysqlTableList** permet de lister les noms des tables MySQL grâce aux commandes SELECT et WHERE.	
Basic settings	*Component list*	Sélectionnez le composant **tMysqlConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Where clause for table name selection	Saisissez la commande WHERE permettant d'identifier les tables sur lesquelles effectuer l'opération d'itération.
Utilisation	Il faut utiliser ce composant en association avec les autres composants MySQL, notamment avec le **tMysqlConnection**.	
Limitation	n/a	

Scénario : Effectuer une opération d'itération sur des tables et en effacer le contenu grâce à un modèle SQL défini par l'utilisateur (SQL Template)

Le scénario Java suivant décrit un Job à trois composants qui permet d'effectuer une opération d'itération, dans les tables spécifiées d'une base de données MySQL, en utilisant une clause de type WHERE. Le Job permet ensuite d'effacer le contenu de ces tables directement sur le SGBD grâce à un modèle SQL (SQL Template) défini par l'utilisateur.

Pour une utilisation avancée, commencez par créer une connexion à la base de données qui contient les tables que vous voulez vider de leur contenu.

- Dans la vue en arborescence du **Repository**, développez le noeud **Metadata** puis cliquez droit sur **DB Connections** pour créer une connexion à la base de données de travail et stocker les informations de connexion en local.
  Pour plus d'informations sur les métadonnées (metadata), consultez le chapitre *Centraliser les métadonnées* du **Guide Utilisateur** de **Talend Open Studio**.
  Sinon, déposez un composant **tMySQLConnection** dans l'éditeur graphique et renseignez les paramètres de connexion manuellement.
- A partir du **Repository**, déposez dans l'éditeur graphique la connexion que vous venez de créer.
  La boîte de dialogue **[Components]** s'affiche à l'écran.

- Sélectionnez le composant **tMySQLConnection** puis cliquez sur **OK**.
  Le composant **tMySQLConnection** apparait alors dans l'éditeur graphique et les paramètres de connexion, accessibles depuis la vue Basic settings, sont automatiquement renseignés.
- A partir de la Palette, cliquez-déposez les composants **tMysqlTableList** et **tELT** dans l'éditeur graphique.
- Reliez les composants **tMysqlConnection** et **tMysqlTableList** à l'aide d'un lien Trigger de type **OnSubJobOk**.

- Reliez les composants **tMysqlTableList** et **tELT** à l'aide d'un lien de type **Iterate**.
- Double-cliquez sur le **tMysqlConnection** pour afficher la vue **Basic settings** du composant si vous voulez en vérifier les paramètres de connexion.

Dans cet exemple, on veut se connecter à une base de données MySQL appelée *examples*.

- Dans l'éditeur graphique, double-cliquez sur le composant **tMysqlTableList** pour en afficher les paramètres de bases et les définir (vue **Basic settings**).

- Sélectionnez la connexion MySQL appropriée à partir de la liste déroulante **Component list**, dans le cas où vous utilisez plus d'une connexion.

- Renseignez la clause de type WHERE dans le champ approprié, en veillant à utiliser la bonne syntaxe, pour effectuer une itération sur la ou les tables dont vous voulez effacer le contenu. Dans ce scénario, on veut opérer l'itération sur toutes les tables dont le nom commence par "ex".
- Dans l'éditeur graphique, double-cliquez sur le composant **tELT** pour en afficher les paramètres de base et les définir (vue **Basic settings**).

- Dans le champ **Database Name**, renseignez le nom de la base de données où se trouvent les tables sur lesquelles vous voulez travailler.
- Sélectionnez la connexion MySQL appropriée à partir de la liste déroulante **Component list**, dans le cas où vous utilisez plus d'une connexion.
- Cliquez dans le champ **Table name** et appuyez sur **Ctrl+Espace** pour atteindre la liste générale des variables.
- A partir de cette liste de variables, sélectionnez *((String)globalMap.get("tMysqlTableList_1_CURRENT_TABLE"))*.

Comment créer votre propre modèle SQL (SQL Template) :

- Dans la vue en arborescence du **Repository**, développez successivement les noeuds **SQL Templates** et **MySQL**.

- Cliquez-droit sur le noeud **UserDefined** et sélectionnez l'option **Create SQLTemplate** à partir de la liste.
  L'assistant **New SQLTemplate** s'affiche à l'écran.

- Saisissez un nom pour le nouveau modèle (SQL template) et remplissez les autres champs si nécessaire, puis cliquez sur **Finish** pour fermer l'assistant.
  L'éditeur de modèle SQL s'affiche dans l'éditeur graphique.
- Supprimez le code existant et saisissez le code permettant d'exécuter l'action voulue, c'est-à-dire, ici, effacer le contenu de toutes les tables dont les noms commencent par "ex".

Dans un code SQL template, il est impératif d'utiliser le nom exact de la variable, en association avec le paramètre désignant le nom de la table ("__TABLE-NAME__" dans cet exemple).
Pour afficher le nom de la variable utilisée, positionnez votre curseur sur le champ **Table Name** de la vue Basic Settings du composant **tELT**.

- Appuyez sur **Ctrl+S** pour sauvegarder le nouveau modèle que vous venez de définir.

La prochaine étape consiste à ajouter ce nouveau modèle SQL template à la liste de modèles du même type dans le composant **tELT**.

Comment ajouter votre propre modèle SQL à la liste de modèles du même type :

- Dans la vue **Component** du composant **tELT**, cliquez sur l'onglet **SQL Templates** pour afficher la liste de modèles **SQLTemplate List**.

- Cliquez sur le bouton **Add** et ajoutez deux lignes de modèles SQL template.
- Cliquez sur la première ligne pour afficher un menu déroulante puis cliquez sur la flèche de ce menu pour dérouler la liste des modèles.

- Dans la liste, sélectionnez le modèle SQL template que vous venez de créer.
- Assurez-vous que le modèle de la seconde ligne est bien en mode **Commit**.
- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

Toutes les tables de la base de données MySQL `examples` dont le nom commence par "ex" sont alors vidées de leur contenu.

Scénario associé

Pour un scénario associé, consultez le *Scénario : Itérer une table de base de données et lister le nom des colonnes de la table, page 466*.

tNetezzaBulkExec

Propriétés du tNetezzaBulkExec

Famille de composant	Databases/Netezza	
Fonction	Le composant **tNetezzaBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tNetezzaBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données Netezza.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tNetezzaConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.

	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit exister pour que l'opération d'Insert soit autorisée.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant (**Repository**).
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	File Name	Nom du fichier à traiter. Voir également :*Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
Advanced settings	*Field Separator*	Caractère, chaîne ou expression régulière séparant les champs.
	Require quotes (") *around data files*	Cochez cette case pour utiliser des caractères pour entourer les données.
	Row Separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Escape character	Caractère d'échappement de la ligne.
	Date format / Date delimiter	**Date format** *:* Utilisez ce champ pour représenter l'ordre des mois, des jours et des années sous forme de chaîne de caractères. **Date delimiter** : Utilisez ce champ pour définir l'élément séparateur entre les valeurs.
	Time format/ Time delimiter	**Time format** *:* Utilisez ce champ pour représenter le format de l'heure sous forme de chaîne de caractères. **Time delimiter** : Utilisez ce champ pour définir l'élément séparateur entre les valeurs.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	Max Errors	Saisissez le nombre maximal d'erreurs possibles avant que le processus ne s'arrête.
	Skip Rows	Renseignez le nombre de lignes à ignorer.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est principalement utilisé lorsque des transformations non particulières sont requises sur les données à charger dans la base de données Netezza.	
Limitation	n/a	

Scénarios associés

Pour des scénarios associés au composant **tNetezzaBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tNetezzaClose

Propriétés du tNetezzaClose

Famille de composant	Databases/Netezza	
Fonction	Le composant **tNetezzaClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données Netezza.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tNetezzaConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tNetezzaConnection** et **tNetezzaRollback**. Il est généralement utilisé avec un composant **tNetezzaConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tNetezzaClose**, consultez *tMysqlConnection, page 471*

tNetezzaCommit

Propriétés du tNetezzaCommit

Ce composant est étroitement lié aux composants **tNetezzaConnection** et **tNetezzaRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Netezza	
Fonction	Le composant **tNetezzaCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tNetezzaConnection** dans la liste.
	Close Connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tNetezzaCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants Netezza et notamment avec les composants **tNetezzaConnection** et **tNetezzaRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tNetezzaConnection** et **tNetezzaRollback**. Ils sont généralement utilisés avec un composant **tNetezzaConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tNetezzaCommit,** consultez le *Scénario : Insérer des données dans des tables mère/fille, page 471*.

tNetezzaConnection

Propriétés du tNetezzaConnection

Ce composant est étroitement lié aux composants **tNetezzaCommit** et **tNetezzaRollback**. Ils sont généralemet utilisés avec un composant **tNetezzaConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/Netezza	
Fonction	Le composant **tNetezzaConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Table Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Additional JDBC Parameters	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Utilisation	Ce composant est généralement utilisé avec des composants Netezza, notamment les composants **tNetezzaCommit** et **tNetezzaRollback**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au composant **tNetezzaConnection**, consultez le *Scénario : Insérer des données dans des tables mère/fille, page 471*.

tNetezzaInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tNetezzaInput

Famille de composant	Databases/Netezza	
Fonction	Le composant **tNetezzaInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tNetezzaInput** exécute une requête de base de données dans laquelle l'ordre doit strictement correspondre à l'ordre défini dans le schéma. Puis la liste des champs est transmise au composant via une connexion Main row.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionne le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case lorsque vous utilisez un composant **tOracleConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du Guide utilisateur de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du Guide utilisateur de **Talend Open Studio**.

	Table name	Nom de la table de base de données.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Use cursor*	Cochez cette case et définissez le nombre de lignes avec lesquelles vous souhaitez travailler en une fois. Cette option permet d'optimiser les performances.
	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimer les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Netezza.	

Scénarios associés

Consultez les scénarios du composant **tNetezzaInput** :

- le *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239*.
- le *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241*.

Pour un scénario associé, consultez le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du composant **tContextLoad**.

tNetezzaNzLoad

Ce composant invoque l'utilitaire nzload de Netezza afin d'insérer des enregistrements dans une base de données Netezza. Ce composant peut être utilisé soit en mode standalone, il chargera alors les données à partir d'un fichier de données existant, soit connecté à un flux d'entrée, afin de charger des données à partir du composant connecté.

Propriétés du tNetezzaNzLoad

Famille de composant	Databases/Netezza	
Fonction	Le composant **tNetezzaNzLoad** insère des données dans la table d'une base de données Netezza via l'utilitaire nzload de Netezza.	
Objectif	Ce composant permet de charger en masse des données dans une table Netezza soit à partir d'un fichier, soit à partir d'un flux d'entrée, soit à partir d'un tube nommé.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table dans laquelle les données doivent être insérées.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.

		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Data file	Chemin d'accès complet au fichier de données à utiliser. Si ce composant est utilisé seul (non connecté à un autre composant par un flux d'entrée), alors saisissez le nom d'un fichier existant à charger dans la base de données. S'il est connecté à un autre composant par un flux d'entrée, saisissez le nom du fichier à générer et à écrire avec les données d'entrée, afin de l'utiliser plus tard avec nzload pour charger dans la base de données.
	Use named-pipe	Cochez cette case afin d'utiliser un tube nommé à la place d'un fichier de données. Cette option ne peut être utilisée que lorsque le composant est connecté à un autre composant par un flux d'entrée. Quand la case est cochée, aucun fichier de données n'est généré, et les données sont transférées à l'utilitaire nzload via un tube nommé. Cette option améliore grandement les performances sous Linux et Windows. Ce composant, en mode tube nommé, utilise une interface JNI pour créer un tube nommé et écrire dedans, dans toute plateforme Windows. Le chemin d'accès associé au fichier DLL JNI doit être configuré dans le chemin d'accès à la bibliothèque Java. Le composant est automatiquement fourni dans le Studio avec deux DLL pour les systèmes d'exploitation de 32 et 64 bits.
	Named-pipe name	Spécifiez un nom pour le tube nommé à utiliser. Vérifiez que le nom saisi est valide.
Advanced settings	*Use existing control file*	Cochez cette case pour fournir un fichier de contrôle à utiliser avec l'utilitaire nzload au lieu de spécifier explicitement toutes les options dans le composant. Lorsque cette case est cochée, l'option **Data file** ainsi que les autres options liées au nzload ne s'appliquent plus. Veuillez vous référer au manuel du nzload de Netezza pour plus d'informations concernant la création d'un fichier de contrôle.
	Control file	Saisissez le chemin d'accès au fichier de contrôle à utiliser, entre guillemets doubles, ou cliquez sur le bouton **[...]** afin de parcourir votre répertoire jusqu'au fichier de contrôle. Cette option est passé à l'utilitaire nzload via l'argument *-cf*.

	Field separator	Caractère, chaîne de caractères ou expression régulière utilisé pour séparer les champs. *Argument delim de l'utilitaire nzload. Si vous n'utilisez pas l'option* **Wrap quotes around fields**, *vous devez être sûr que le délimiteur n'est pas inclus dans les données insérées dans la base de données. La valeur par défaut est* `\t` *ou* `TAB`*.. Pour améliorer les performances, utilisez la valeur par défaut.*
	Wrap quotes around fields	Cette option ne s'applique qu'aux colonnes de type *String*, *Byte*, *Byte[]*, *Char*, et *Object*. Sélectionnez soit : **None** : n'entourez pas les valeurs des colonnes de guillemets. **Single quote** : entourez les valeurs des colonnes de guillemets simples. **Double quote** : entourez les valeurs des colonnes de guillemets doubles. *Si vous utilisez l'option* **Single quote** *ou* **Double quote**, *il est nécessaire d'utiliser* \ *en tant que* **Escape char**.
	Advanced options	Définissez les arguments nzload dans le tableau correspondant. Cliquez sur le bouton **[+]** autant de fois que nécessaire afin d'ajouter les arguments au tableau. Cliquez sur le champ **Parameter** et choisissez l'argument souhaité dans la liste, puis cliquez sur le champ **Value** correspondant et saisissez une valeur entre guillemets.
Parameter	*-If*	Nom du fichier de log à générer. Les logs seront ajoutés à la suite du fichier de log, s'il existe déjà. Si le paramètre n'est pas spécifié, le nom par défaut du fichier de log est '*<table_name>.<db_name>.nzlog*'. Il est généré dans le répertoire de travail courant dans lequel le Job est exécuté.
	-bf	Nom du fichier de mauvais enregistrements à générer. Le fichier de mauvais enregsitrements contient tous les enregistrements ne pouvant être chargés, à cause d'une erreur interne Netezza. Les enregistrements seront ajoutés à la suite si le fichier de mauvais enregistrements existe déjà. Si le paramètre n'est pas spécifié, le nom par défaut du fichier de mauvais enregistrements est '*<table_name>.<db_name>.nzbad*'. Il est généré dans le répertoire de travail courant, dans lequel le Job est exécuté.
	-ouputDir	Chemin d'accès au répertoire dans lequel les fichiers de log et de mauvais enregistrements sont générés. Si le paramètre n'est pas spécifié, les fichiers seront générés sous le répertoire courant dans lequel le Job est exécuté.

	-logFileSize	Taille maximale du fichier de log. La valeur est en MB. La valeur par défaut est de 2000 ou 2GB. Pour économiser de l'espace disque, spécifiez un petite valeur si votre Job est souvent exécuté.
	-compress	Sélectionnez cette option si le fichier de données est compressée. Les valeurs valides sont "**TRUE**" et "**FALSE**". La valeur par défaut est "**FALSE**". Cette option n'est valide que si ce composant est utilisé par lui-même et non connecté à un autre composant via un flux d'entrée.
	-skipRows <n>	Nombre de lignes à ignorer au début du fichier de données. Paramétrez la valeur à "1" si vous souhaitez ignorer la ligne d'en-tête du fichier d'entrée. La valeur par défaut est "0". Cette option ne doit être sélectionnée que si le composant est utilisé par lui-même et non connecté à un autre composant via un flux d'entrée.
	-maxRows <n>	Nombre de lignes à charger à partir du fichier d'entrée. Cette option ne doit être sélectionnée que si le composant est utilisé par lui-même et non connecté à un autre composant via un flux d'entrée.
	-maxErrors	Nombre maximal d'enregistrements d'erreurs à autoriser avant de terminer le processus de chargement. La valeur par défaut est "1".
	-ignoreZero	Les zéros contenus dans une source binaire génèreront des erreurs. Paramétrez cette option à "NO" pour générer des erreurs ou à "YES" pour ignorer les bits ayant la valeur zéro. La valeur par défaut est "NO".
	-requireQuotes	Cette option nécessite que toutes les valeurs soient entourées de guillemets. La valeur par défaut est "FALSE". Cette option ne fonctionne pas avec un flux d'entrée pour le moment. Utilisez cette option uniquement en mode standalone avec un fichier existant.
	-nullValue <token>	Spécifiez le jeton pour indiquer une valeur null dans le fichier de données. La valeur par défaut est "NULL". Afin d'améliorer légèrement les performances, vous pouvez configurer cette valeur comme un champ vide en spécifiant la valeur : "\'\'".

	-fillRecord	Traite les derniers champs d'entrée manquants comme des null. Vous n'avez pas besoin de spécifier de valeur pour cette option dans le champ du tableau. Cette option n'est pas activée par défaut, les champs d'entrée devront donc correspondre exactement à toutes les colonnes de la table par défaut. Les derniers champs d'entrée doivent être nul-lables dans la base de données.
	-ctrlChar	Accepte les caractères de contrôle dans les champs char/varchar (vous devrez échapper les caractères NUL, CR et LF). Vous n'avez pas besoin de spécifier de valeur pour cette option dans le champ de valeur du tableau. Cette option est désactivée par défaut.
	-ctInString	Accepte caractère CR non échappé dans des champs char/varchar (LF becomes only end of row). Vous n'avez pas besoin de spécifier de valeur pour cette option dans le champ de valeur du tableau. Cette option est désactivée par défaut.
	-truncString	Tronque toute valeur de chaîne de caractères qui dépasse sa valeur déclarée de stockage char/varchar. Vous n'avez pas besoin de spécifier de valeur pour cette option dans le champ de valeur du tableau. Cette option est désactivée par défaut.
	-dateStyle	Spécifiez le format de la date des données d'entrée. Les valeurs valides sont : "YMD", "Y2MD", "DMY", "DMY2", "MDY", "MDY2", "MONDY", "MONDY2". La valeur par défaut est "YMD". Le format de date dans la colonne du schéma du composant doit correspondre à la valeur spécifiée ici. Par exemple, si vous souhaitez charger une colonne DATE, spécifiez ainsi le format de date dans le schéma du composant : "yyyy-MM-dd" et paramétrez l'option -dateStyle à "YMD". Pour une description plus précise du chargement de champs de dates et de temps, consultez la section *Charger des colonnes de type DATE, TIME et TIMESTAMP, page 562*.
	-dateDelim	Caractère délimiteur entre les parties d'une date. La valeur par défaut est "-" pour tous les styles de date, sauf pour "MONDY[2]" pour lequel elle est " " (espace vide). Le format de date dans la colonne du schéma du composant doit correspondre à la valeur spécifiée ici.
	-y2Base	Première année exprimable via l'utilisation du style de date à deux chiffres (Y2).

	-timeStyle	Spécifiez le format du temps dans les données d'entrée. Les valeurs valides sont : "24HOUR" et "12HOUR". La valeur par défaut est "24HOUR". Pour améliorer légèrement les performances, gardez la valeur par défaut. Le format de temps dans la colonne du schéma du composant doit correspondre à la valeur spécifiée ici. Par exemple, si vous souhaitez charger une colonne de type TIME, configurez le format de date dans le schéma du composant à "HH:mm:ss" et l'option -timeStyle à "24HOUR". Pour une description plus précise du chargement de champs de dates et de temps, consultez la section *Charger des colonnes de type DATE, TIME et TIMESTAMP, page 562*.
	-timeDelim	Caractère délimiteur entre les parties d'une donnée de temps. La valeur par défaut est ":". Le format de temps dans la colonne du schéma du composant doit correspondre à la valeur spécifiée ici.
	-timeRoundNanos	Autorise mais arrondit les nombres différents de zéro plus petits qu'une microseconde.
	-boolStyle	Spécifiez le format dans lequel les données de type Boolean sont écrites dans les données. Les valeurs valides sont : "1_0", "T_F", "Y_N", "TRUE_FALSE", "YES". La valeur par défaut est "1_0". Pour améliorer légèrement les performances, gardez la valeur par défaut.
	-allowRelay	Permet de continuer le chargement malgré une ou plusieurs réinitialisation(s) ou échec(s) des SPU (Snippet Processing Unit). Par défaut, ce comportement n'est pas autorisé.
	-allowRelay <n>	Spécifiez le nombre de continuations du chargement autorisé. La valeur par défaut est "1".
	Encoding	Sélectionnez l'encodage à partir de la liste.
	Specify nzload path	Cochez cette case afin de spécifier le chemin d'accès complet à l'exécutable nzload. Vous devez cocher cette case si le chemin d'accès n'est pas spécifié dans la variable d'environnement PATH.
	Full path to nzload executable	Chemin d'accès complet à l'exécutable nzload sur la machine en cours d'utilisation. Il est recommandé de spécifier le chemin d'accès dans la variable d'environnement PATH au lieu de sélectionner cette option.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données Netezza. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject**s filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.

Charger des colonnes de type DATE, TIME et TIMESTAMP

Lorsque ce composant est utilisé avec un flux d'entrée, le format de date spécifié dans le schéma du composant doit correspondre à la valeur spécifiée pour les options -dateStyle, -dateDelim, -timeStyle, et -timeDelim. Veuillez vous référer aux exemples suivants :

Type de données	Format de date dans le schéma	-dateStyle	-dateDelim	-timeStyle	-timeDelim
DATE	"yyyy-MM-dd"	"YMD"	"-"	n/a	n/a
TIME	"HH:mm:ss"	n/a	n/a	"24HOUR"	":"
TIMESTAMP	"yyyy-MM-dd HH:mm:ss"	"YMD"	"-"	"24HOUR"	":"

Scénario associé

Pour un scénario associé, consultez le *Scénario : Insérer des données dans une base MySQL* du composant **tMysqlOutputBulkExec**.

tNetezzaOutput

Propriétés du tNetezzaOutput

Famille de composant	Databases/Netezza	
Fonction	Le composant **tNetezzaOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tNetezzaOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Use an existing connection	Cochez cette case lorsque vous utilisez un composant **tNetezzaConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.

	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.
	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondant au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.

		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du Guide utilisateur de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du Guide utilisateur de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Extend Insert	Cochez cette case pour insérer un ensemble de lignes définies au lieu d'insérer les lignes une par une. Cette option permet un important gain de performance. **Number of rows per insert**: : saisissez le nombre de lignes à insérer en un bloc. Notez que si vous sélectionnez un nombre important de lignes, cela peut augmenter la quantité de mémoire utilisée et donc diminuer les performances.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tNetezzaOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.

	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données Netezza. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour un scénario associé au composant **tNetezzaOutput**, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput.**
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMysqlOutput**.

tNetezzaRollback

Propriétés du tNetezzaRollback

Ce composant est étroitement lié aux composants **tNetezzaCommit** et **tNetezzaConnection**. Ils sont généralement utilisés ensemble lors d'une transaction.

Famille de composant	Databases/Netezza	
Fonction	Le **tNetezzaRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant permet d'éviter le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant **tNetezzaConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce component est généralement utilisé avec d'autres composants Netezza, notamment les composants **tNetezzaConnection** et **tNetezzaCommit** components.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au composant **tNetezzaRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522*.

tNetezzaRow

Propriétés du tNetezzaRow

Famille de composant	Databases/Netezza	
Fonction	**tNetezzaRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tNetezzaRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tNetezzaConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.

	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à traiter.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Propagate QUERY's recordset	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.

	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des “?” dans l’instruction SQL définie dans le champ **Query** de l’onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l’instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l’écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d’exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénario associé

Pour un scénario associé au composant **tNetezzaRow**, consultez le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526*.

tOracleBulkExec

Propriétés du tOracleBulkExec

Les composants **tOracleOutputBulk** et **tOracleBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tOracleOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le changement des données dans la base de données.

Famille de composant	Databases/Oracle	
Fonction	Le composant **tOracleBulkExec** insère, ajoute, remplace ou supprime les données d'une base de données Oracle.	
Objectif	Le **tOracleBulkExec** est un composant dédié, il permet un gain de performance lors du traitement des données de la base de données Oracle.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez un composant **tOracleConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Connection type	Liste déroulante contenant les pilotes disponibles
	DB Version	Sélectionnez la version d'Oracle que vous utilisez

	Host	Adresse IP du serveur de base de données
	Port	Numéro du port d'écoute du serveur de base de données
	Database	Nom de la base de données.
	Schema	Nom du schéma.
Perl uniquement	*Service Name*	Nom du service ou SID de la base de données Oracle. Dans les projets Java, les informations de connexion doivent être complètes.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Data file name	Nom du fichier à traiter. Voir également :*Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Insére des lignes dans une table vide. Le Job s'arrête en cas de doublons. **Update** : Met à jour les entrées existantes. **Append** : Ajoute des lignes aux données déjà existantes. **Replace** : Ecrase des lignes de la table. **Truncate** : Supprimer les entrées correspondant au flux d'entrée. *Il est nécessaire de spécifier au minimum une clé de recherche sur laquelle baser les opérations d'Update. Il est possible de définir les colonnes qui agiront comme clé de recherche à partir du schéma, pour une utilisation de base, ou à partir des options avancées (Advanced settings) pour une utilisation optimisée de ces opérations.*

	Schema et *Edit schema*	Dans une procédure stockée, le schéma est un paramètre d'entrée. Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma soit local (built-in) soit distant dans le **Repository**.
Advanced settings	*Advanced separator (for numbers)*	Cochez cette option pour modifier les séparateurs utilisés pour les nombres : **Thousands separator** : définissez le séparateur utilisé pour les milliers. **Decimal separator** : définissez le séparateur utilisé pour les décimaux.
	Use existing control file	Cochez cette case si vous utilisez un fichier de contrôle (.ctl) et spécifiez son chemin d'accès dans le champ **.ctl file name**.
	Record format	Définissez le format de l'enregistrement : **Default** : les paramètres du format sont définis par défaut. **Stream** : déterminez le séparateur de fin d'enregistrement. **Fixed** : déterminez la longueur d'enregistrement. **Variable** : déterminez la taille du champ spécifiant la longueur de l'enregistrement.
	Specify .ctl file's INTO TABLE clause manually	Cochez cette case pour renseigner manuellement la clause INTO TABLE du fichier de contrôle, directement dans le code.
	Fields terminated by	Caractère, chaîne ou expression régulière séparant les champs : **None** : aucun caractère de séparation. **Whitespace** : le caractère de séparation est un espace. **EOF (used for loading LOBs from lobfile)** : la séparation est constituée d'un caractère de fin de fichier (End Of File). **Other terminator** : la séparation est constituée d'un caractère autre que ceux précédemment énoncés.
	Use fields enclosure	Cochez cette case si vous voulez utiliser des caractères pour entourer le texte : **Fields enclosure (left part)** : caractère délimitant les champs sur la gauche. **Field enclosure (right part)** : caractère délimitant les champs sur la droite.
	Use schema's Date Pattern to load Date field	Cochez cette case pour utiliser le modèle de date contenu dans le schéma pour remplir le champ concernant la date.
	Specify field condition	Cochez cette case pour définir une condition de chargement des données.
	Preserve blanks	Cochez cette case pour conserver les espaces vides.
	Trailing null columns	Cochez cette case pour charger des données avec des colonnes vides.

	Load options	Cliquez sur le bouton [+] afin d'ajouter des options de chargement des données : **Parameter** : choisissez l'un des paramètres de chargement à partir de la liste déroulante. **Value** : définissez la valeur à attribuer au paramètre.
	NLS Language	A partir de la liste déroulante, définissez la langue utilisée pour les données qui ne sont pas codées en Unicode.
	Set Parameter NLS_TERRITORY	Cochez cette case pour modifier les conventions d'appellation utilisées pour la numérotation des jours et des semaines. La valeur par défaut est celle du système d'exploitation.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	Output	Sélectionnez la sortie à utiliser pour récupérer la sortie standard de la base Oracle : **to console** : vers la console. **to global variable** : vers la variable globale.
	Convert columns and table names to uppercase	Cochez cette case pour mettre le nom des colonnes et celui de la table en majuscules.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Perl uniquement	*Fields terminated by*	Caractère, chaîne ou expression régulière séparant les champs.
Perl uniquement	*Fields optionnally enclosed by*	Caractères entourant les données.
Perl uniquement	*Encoding*	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données d'une base de données.
Utilisation	Ce composant dédié offre performance et flexibilité lors de requêtes sur les bases de données Oracle.	

Scénario : Supprimer et insérer des données dans une base Oracle

Ce scénario décrit comment supprimer le contenu d'une base de données Oracle et d'insérer de nouvelles données à partir du contenu d'un fichier d'entrée. Le Job est constitué de trois composants qui créent le contenu, le transfert dans un fichier qui sera ensuite chargé dans la base Oracle après que les données de la table aient été supprimées.

- Cliquez et déposez les composants suivants : **tOracleInput**, **tFileOutputDelimited**, **tOracleBulkExec**.
- Connectez le composant **tOracleInput** au **tFileOutputDelimited** à l'aide d'un lien **Row main**.
- Et connectez le **tOracleInput** au **tOracleBulkExec** à l'aide d'un lien **OnSubjobOk**.
- Paramétrez les informations de connexion à la base de données Oracle. Nous vous conseillons de stocker ces informations dans une métadonnée du **Repository** pour pouvoir les réutiliser par la suite dans tous vos Job.

- Paramétrez le schéma, s'il n'est pas conservé dans le **Repository**. Dans cet exemple, le schéma est comme suit : *ID_Contract*, *ID_Client*, *Contract_type*, *Contract_Value*.
- Dans le champ **Encoding**, sélectionnez l'option **Custom** et définissez *AL32UTF8* comme encodage.
- Dans le champ **Basic settings** du composant **tFileOutputDelimited**, paramétrez ses propriétés, notamment les champs **File Name**, **Row separator** et **Fields separator** avec respectivement le chemin d'accès et le nom du fichier, le séparateur de lignes et de champs.
- Dans le champ **Encoding**, saisissez le même encodage que celui saisi dans les propriétés du composant **tOracleInput**.
- Double-cliquez ensuite sur le composant **tOracleBulkExec** pour définir les propriétés d'alimentation de la base de données.

- Dans le champ **Property Type**, sélectionnez le mode **Repository** si vous avez stocké les informations de connexion à la base de données sous le noeud **Metadata** du **Repository** ou sélectionnez le mode **Built-In** pour les définir manuellement. Dans ce scénario, on utilise le mode **Built-In**.
- Définissez si nécessaire les paramètres de connexion dans les champs **Host**, **Port**, **Database**, **Schema**, **Username** et **Password**.
- Dans le champ **Table**, renseignez le nom de la Table à alimenter et dans le champ **Action on data**, sélectionnez l'action à effectuer, ici : **Insert**
- Dans le champ **Schema**, sélectionnez le mode **Built in** puis cliquez sur le bouton **[...]** à côté du champ **Edit Schema** pour définir les données à passer au composant suivant.
- Cliquez sur la vue **Advanced settings** pour configurer les paramètres avancés du composant.

- Cochez la case **Use an existing control file** si vous avez stocké les informations relatives à l'état de la base de données dans un fichier de contrôle (.ctl). Sinon, renseignez si nécessaire les champs suivants manuellement : **Record format**, **Specify .ctl file's INTO TABLE clause manually**, **Field terminated by**, **Use field enclosure**, **Use schema's Date Pattern to load Date field**, **Specify field condition**, **Preserve blanks**, **Trailing null columns**, **Load options**, **NLS Language** et **Set Parameter NLS_TERRITORY** en fonction de votre base de données.
- Dans le champ **Encoding Type**, paramétrez l'encodage comme dans les étapes précédentes.
- Et dans le champ **Output**, sélectionnez l'option **to console**, ainsi la sortie standard de la base sera affichée dans la console.

Appuyez sur **F6** pour exécuter le Job. La sortie de log apparaît dans la vue **Run** et la table est alimentée par les données du fichier paramétré.

Voir également : *Scénario : Insérer des données dans une base MySQL, page 520*

tOracleClose

Propriétés du tOracleClose

Famille de composant	Databases/Oracle	
Fonction	Le composant **tOracleClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données Oracle.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tOracleConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tOracleConnection** et **tOracleRollback**. Il est généralement utilisé avec un composant **tOracleConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tOracleClose**, consultez *tMysqlConnection, page 471*

tOracleCommit

Propriétés du tOracleCommit

Ce composant est étroitement lié aux composants **tOracleCommit** et **tOracleRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Oracle	
Fonction	Le composant **tOracleCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tOracleConnection** dans la liste.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tOracleCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants Oracle et notamment **tOracleConnection** et **tOracleRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tOracleConnection** et **tOracleRollback**. Ils sont généralement utilisés avec un composant **tOracleConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tOracleCommit**, consultez *tMysqlConnection, page 471*

tOracleConnection

Propriétés du tOracleConnection

Ce composant est étroitement lié aux composants **tOracleCommit** et **tOracleRollback**. Ils sont généralement utilisés avec un composant **tOracleConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/Oracle	
Fonction	Le composant **tOracleConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Connection type	Liste déroulante des pilotes disponibles.
	DB Version	Sélectionnez la version d'Oracle que vous utilisez.
	Use tns file	Cochez cette case pour utiliser les métadonnées d'un contexte contenu dans un fichier tns. Un même fichier Tns peut contenir plusieurs contextes. **TNS File** : Renseignez manuellement le chemin d'accès ou cliquez sur le bouton **[...]** pour parcourir vos dossiers jusqu'au fichier Tns à utiliser. **Select a DB Connection in Tns File** : Cliquez sur le bouton **[...]** pour afficher tous les contextes du fichier Tns et sélectionner celui à utiliser.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.

	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Utilisation	Ce composant est généralement utilisé avec des composants Oracle, notamment les composants **tOracleCommit** et **tOracleRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tOracleCommit** et **tOracleRollback**. Ils sont généralement utilisés avec un composant **tOracleConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tOracleConnection**, consultez *tMysqlConnection, page 471*

tOracleInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tOracleInput

Famille de composant	Databases/Oracle	
Fonction	Le composant **tOracleInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tOracleInput** exécute une requête de base de données dans laquelle l'ordre doit strictement correspondre à l'ordre défini dans le schéma. Puis la liste des champs est transmise au composant via une connexion Main row.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionne le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case lorsque vous utilisez un composant **tOracleConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Connection type	Liste déroulante des pilotes disponibles.
	DB Version	Sélectionnez la version d'Oracle que vous utilisez.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Oracle schema	Nom du schéma Oracle.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.

		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table name	Nom de la table de base de données.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
	Use cursor	Cochez cette case et définissez le nombre de lignes avec lesquelles vous souhaitez travailler en une fois. Cette option permet d'optimiser les performances.
	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimer les espaces en début et en fin de champ dans les colonnes sélectionnées.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Oracle.	

Scénarios associés

Consultez les scénarios du composant **tOracleInput** :

- le *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239*.
- le *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241*.

Pour un scénario associé, consultez le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du composant **tContextLoad**.

tOracleOutput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tOracleOutput

Famille de composant	Databases/Oracle	
Fonction	Le composant **tOracleOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tOracleOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tOracleConnection**. Lorsque vous la décochez, une case (cochée par défaut, suivie d'un champ) apparaît dans les **Advanced settings**, **Batch Size**, qui vous permet de saisir le nombre de lignes dans chaque lot de traitement de données. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Connection type	Liste déroulante des pilotes disponibles.
Java uniquement	*DB Version*	Sélectionnez la version d'Oracle que vous utilisez.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
Perl uniquement	*Oracle schema*	Nom du schéma Oracle
	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.

	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.
	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondant au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).* La fonction de schéma dynamique peut être utilisée dans différents modes : **Insert**, **Update**, **Insert or Update**, **Update or insert**, **Delete**.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
Java uniquement	*Die on error*	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC parameters* *Java uniquement*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
Perl uniquement	*Override any existing NLS_LANG environment variable*	Cochez cette case afin d'écraser les variables précédemment définies dans un environnement au langage NLS.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.

		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tOracleOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
Java uniquement	*Use field options*	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
Java uniquement	*Use Hint Options*	Cochez cette case pour ajouter des indicateurs (ou Hints) permettant d'optimiser le plan d'exécution de la requête afin d'obtenir les meilleurs performances possibles.
Java uniquement	*Convert columns and table to uppercase*	Cochez cette case pour passer les noms de colonnes et de tables en majuscule.
Java uniquement	*Enable debug mode*	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
Java uniquement	*Use Batch Size*	Nombre de lignes dans chaque lot de traitement de données. *Si cette case n'apparaît pas, vérifiez que la case* **Use an existing connection** *dans les* **Basic settings** *est décochée.*
Java uniquement	*Support null in "SQL WHERE" statement*	Cochez cette case pour prendre en compte les valeurs Null lors d'une instruction WHERE.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Ce composant doit être utilisé en tant que composant de sortie. Il permet de faire des actions sur une table ou les données d'une table d'une base de données Oracle. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour un scénario associé au composant **tOracleOutput**, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput.**

- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMysqlOutput**.
- le *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507* du composant **tMysqlOutput**.

tOracleOutputBulk

Propriétés du tOracleOutputBulk

Les composants **tOracleOutputBulk** et **tOracleBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tOracleOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le chargement des données dans la base de données.

Famille de composant	Databases/Oracle	
Fonction	Ecrit un fichier composé de colonnes et basé sur le délimiteur défini et sur les standards Oracle.	
Objectif	Prépare le fichier à utiliser comme paramètre dans la requête INSERT servant à alimenter une base de données Oracle.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Create directory if not exists	Cette option permet de créer le dossier contenant le fichier de sortie s'il n'existe pas déjà.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.

Advanced settings	*Advanced separator (for numbers)*	Cochez cette option pour modifier les séparateurs utilisés pour les nombres : **Thousands separator** : définissez le séparateur utilisé pour les milliers. **Decimal separator** : définissez le séparateur utilisé pour les décimaux.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Row separator	Chaîne (ex : “\n” sous Unix) séparant les lignes.
	Encodind Type	Sélectionnez l’encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec le composant **tOracleBulkExec**. Ensemble ils offrent un gain de performance important pour l’alimentation d’une base de données Oracle.	

Scénarios associés

Pour des scénarios associés au composant **tOracleOutputBulk**, consultez les scénarios suivants :

- *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulk**.
- *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.
- *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tOracleOutputBulkExec

Propriétés du tOracleOutputBulkExec

Les composants **tOracleOutputBulk** et **tOracleBulkExec** sont généralement utilisés ensemble comme deux parties d'un processus en deux étapes. Dans la première étape, un fichier de sortie est généré. Dans la deuxième étape, ce fichier est utilisé lors de l'opération d'INSERT afin de populer une base de données. Cette exécution en deux étapes est unifiée dans le composant **tOracleOutputBulkExec**.

Famille de composant	Databases/Oracle	
Fonction	Le **tOracleOutputBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Ce composant est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données Oracle.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tOracleConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Connection type	Liste déroulante des pilotes disponibles.
	DB Version	Sélectionnez la version d'Oracle que vous utilisez.
	Host	Adresse IP du serveur de base de données.

	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'Insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	File Name	Nom du fichier à traiter. Voir également :*Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Create directory if not exists	Cette case est cochée par défaut. Cette option permet de créer un répertoire où stocker la table de sortie si elle n'existe pas.
	Append	Cochez cette case afin d'ajouter de nouvelles lignes à la fin de l'enregistrement.
	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Truncate** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une clé de recherche sur laquelle baser les opérations d'Update. Il est possible de définir les colonnes qui agiront comme clé de recherche à partir du schéma, pour une utilisation de base, ou à partir des options avancées (Advanced settings) pour une utilisation optimisée de ces opérations.*

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Advanced separator (for numbers)*	Cochez cette option pour modifier les séparateurs utilisés pour les nombres : **Thousands separator** : définissez le séparateur utilisé pour les milliers. **Decimal separator** : définissez le séparateur utilisé pour les décimaux.
	Use existing control file	Cochez cette case si vous utilisez un fichier de contrôle (.ctl).
	Record format	Définissez le format de l'enregistrement : **Default** : les paramètres du format sont définis par défaut. **Stream** : déterminez le séparateur de fin d'enregistrement. **Fixed** : déterminez la longueur d'enregistrement. **Variable** : déterminez la taille du champ spécifiant la longueur de l'enregistrement.
	Specify .ctl file's INTO TABLE clause manually	Cochez cette case pour renseigner manuellement la clause INTO TABLE du fichier de contrôle, directement dans le code.
	Fields terminated by	Caractère, chaîne ou expression régulière séparant les champs : **None** : aucun caractère de séparation. **Whitespace** : le caractère de séparation est un espace. **EOF (used for loading LOBs from lobfile)** : la séparation est constituée d'un caractère de fin de fichier (End Of File). **Other terminator** : la séparation est constituée d'un caractère autre que ceux précédemment énoncés.
	Use fields enclosure	Cohez cette case si vous voulez utiliser des caractères pour entourer le texte : **Fields enclosure (left part)** : caractère délimitant les champs sur la gauche. **Field enclosure (right part)** : caractère délimitant les champs sur la droite.
	Use schema's Date Pattern to load Date field	Cochez cette case pour utiliser le modèle de date contenu dans le schéma pour remplir le champ concernant la date.

	Specify field condition	Cochez cette case pour définir une condition de chargement des données.
	Preserve blanks	Cochez cette case pour conserver les espaces vides.
	Trailing null columns	Cochez cette case pour charger des données avec des colonnes vides.
	Load options	Cliquez sur le bouton **[+]** afin d'ajouter des options de chargement des données : **Parameter** : choisissez l'un des paramètres de chargement à partir de la liste déroulante. **Value** : définissez la valeur à attribuer au paramètre.
	NLS Language	A partir de la liste déroulante, définissez la langue utilisée pour les données qui ne sont pas codées en Unicode.
	Set Parameter NLS_TERRITORY	Cochez cette case pour modifier les conventions d'appellation utilisées pour la numérotation des jours et des semaines. La valeur par défaut est celle du système d'exploitation.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	Output	Sélectionnez dans la liste déroulante l'endroit où rediriger le flux de sortie : **to console** : ver la console de log. **to global variable** : vers une variable globale.
	Convert columns and table names to uppercase	Cochez cette case pour mettre le nom des colonnes et celui de la table en majuscules.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est principalement utilisé lorsqu'aucune transformation particulière n'est requise sur les données à charger dans la base de données.	
Limitation	n/a	

Scénarios associés

Pour des scénarios associés au composant **tOracleOutputBulkExec**, consultez les scénarios suivants :

- *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulk**.
- *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.
- *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tOracleRollback

Propriétés du tOracleRollback

Ce composant est étroitement lié aux composants **tOracleCommit** et **tOracleConnection**. Ils sont généralement utilisés ensemble lors d'une transaction.

Famille de composant	Databases	
Fonction	Le **tOracleRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant **tOracleConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec d'autres composants Oracle, notamment les composants **tOracleConnection** et **tOracleCommit**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tOracleConnection** et **tOracleCommit**. Ils sont généralement utilisés ensemble lors d'une transaction.

Pour un scénario associé au composant **tOracleRollback**, consultez *tMysqlRollback, page 522*.

tOracleRow

Propriétés du tOracleRow

Famille de composant	Databases/Oracle	
Fonction	Le **tOracleRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tOracleRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tOracleConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Connection type	Liste déroulante des pilotes disponibles.
	DB Version	Sélectionnez la version d'Oracle que vous utilisez.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.

	Database	Nom de la base de données.
	Schema	Nom du schéma Oracle.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Table name	Nom de la table de base de données.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Propagate QUERY's recordset	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.

	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des “?” dans l’instruction SQL définie dans le champ **Query** de l’onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l’instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l’écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d’exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénario associé

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l’auto-incrémentation d’une base de données, page 249* du **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du **tMySQLRow**.

tOracleSCD

Le composant **tOracleSCD** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tOracleSCD, page 35*.

tOracleSCDELT

Le composant **tOracleSCDELT** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tOracleSCDELT, page 37*.

tOracleSP

Propriétés du tOracleSP

Famille de composant	Databases/Oracle	
Fonction	Le composant **tOracleSP** appelle une procédure stockée de base de données.	
Objectif	Ce composant permet de centraliser des requêtes multiples ou complexes dans une base de données et de les appeler plus facilement.	
Basic settings	*Use an existing connection*	Cochez cette case et sélectionnez le composant **tOracleConnection** adéquat à partir de la liste **Component** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Connection type	Le type de connexion peut être : - **Oracle SID** - **Oracle Service Name** - **Oracle OCI**
	Property type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier des propriétés du composant. Les champs suivants sont alors pré-remplis.
	DB Version	Sélectionnez la version d'Oracle que vous utilisez.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.

	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification sur l'utilisateur de base de données.
	Schema et *Edit Schema*	Dans une procédure stockée, le schéma est un paramètre d'entrée. Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le **Repository**. Ainsi, il est réutilisable. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	SP Name	Saisissez le nom exact de la Procédure Stockée (SP).
	Is Function / *Return result in*	Cochez cette case, si une seule valeur doit être retournée. Sélectionnez dans la liste la colonne du schéma sur laquelle est basée la valeur à obtenir.
	Parameters	Cliquez sur le bouton **[+]** et sélectionnez dans le champ **Schema Columns** les différentes colonnes nécessaires à la procédure. Notez que le schéma de la SP peut contenir plus de colonnes qu'il n'y a de paramètres utilisés dans la procédure.
		Sélectionnez le **Type** de paramètre : **IN** : paramètre d'entrée (Input) **OUT** : paramètre de sortie (Output)/valeur retournée **IN OUT** : les paramètres d'entrée doivent être retournées sous forme de valeur, même après modifications via la procédure (fonction). **RECORDSET** : les paramètres d'entrée doivent être retournées sous forme d'ensemble de valeurs, au lieu d'une valeur unique. Consultez le composant *tParseRecordSet* si vous voulez analyser un ensemble d'enregistrements d'une table de données ou d'une requête SQL.
		Le type **Custom Type** est utilisé quand une colonne du schéma (**Schema Column**) que vous souhaitez utiliser est définie par l'utilisateur. Deux types de Custom sont disponibles dans le tableau **Parameters**. Pour le premier type : - Cochez la case quand la colonne du schéma correspondant est définie par l'utilisateur. - Si toutes les colonnes du schéma dans le tableau **Parameters** sont de type personnalisé, vous pouvez cocher la case devant **Custom Type**.

		Dans la seconde colonne **Custom Type**, vous pouvez préciser le type de personnalisation. Il peut être - **STRUCT** : utilisé pour un élément. - **ARRAY** : utilisé pour plusieurs éléments.
		Dans la colonne **Custom name**, spécifiez le type de personnalisation. *Dans le tableau* **Parameters**, *si une colonne est de* **Type** *OUT, assurez-vous que son type dans le schéma est bien Object.*
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	NLS Language	A partir de la liste déroulante, définissez la langue utilisée pour les données qui ne sont pas codées en Unicode.
	NLS Territory	Cochez cette case pour modifier les conventions d'appellation utilisées pour la numérotation des jours et des semaines. La valeur par défaut est celle du système d'exploitation.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est un composant intermédiaire. Il peut être utilisé comme composant de début. Dans ce cas, seuls les paramètres d'entrée sont autorisés.	
Limitation	La syntaxe de la Procédure Stockée doit correspondre à celle de la base de données. Quand le paramètre défini dans ce composant est personnalisé (**Custom Type**), les composants de la famille **tJava** doivent être placés avant le **tOracleSP**, afin que les utilisateurs puissent définir des valeurs pour les paramètres personnalisés, ou bien après le **tOracleSP**, pour lire et écrire les paramètres personnalisés de type OUT.	

Scénario : Vérifier le format de numéros à l'aide d'une procédure stockée

Le Job suivant permet de se connecter à une base Oracle contenant des numéros de sécurité social et le nom de leur propriétaire, en appelant une procédure stockée qui vérifie le format des numéros en fonction du format standard ###-##-####. Puis le résultat de la vérifiaction, 1 pour un format valide et 0 pour un format non valide, est affiché en sortie dans la console de la vue **Run**.

- Cliquez et déposez les composants suivants : **tOracleConnection**, **tOracleInput**, **tOracleSP** et **tLogRow**.
- Reliez le composant **tOracleConnection** au **tOracleInput** à l'aide d'un lien **Then Run** puisqu'aucune donnée n'est traitée ici.
- Et connectez les autres composants à l'aide de liens **Row Main** puisque des lignes sont transmises comme paramètres au composant SP et à la console.
- Dans l'onglet **Basic settings** du composant **tOracleConnection**, paramétrez les informations de connexion à la base de données correspondante. Vous pourrez donc réutiliser cette information dans tout autre composant lié aux bases de données.
- Puis double-cliquez sur le composant **tOracleInput** et paramétrez ses propriétés.

- Cochez la case **Use an existing connection** et sélectionnez le composant **tOracleConnection** dans la liste pour réutiliser les informations de connexion déjà paramétrées.
- Sélectionnez l'option **Repository** dans le champ **Property type** puisque le schéma Oracle est défini dans une métadonnée du dossier **Db Connections** du **Repository**. Si vous n'avez pas enregistré de métadonnée contenant les informations de connexion à la base de données Oracle dans le **Repository**, renseignez le nom du schéma manuellement.
- Puis sélectionnez l'option **Repository** dans le champ **Schema**, et récupérez le schéma correspondant à la table de la base de données Oracle.

	ID	NAME	CITY	SSNUMBER
1	1	Jack	LA	123-45-6789
2	2	Tom	NYC	123-A5-6789
3	3	Bill	SF	123=45-6789
4	4	Jana	NYC	236-52-2956
5	5	Brandon	SLC	561-52-B267

- Dans cet exemple, la table contenant le numéro de sécurité social contient un schéma à quatre colonnes dont : *ID*, *NAME*, *CITY* et *SSNUMBER*.
- Dans le champ **Query**, saisissez la requête Select suivante ou sélectionnez-la dans la liste si vous l'aviez stockée dans le **Repository**.
  `select ID, NAME, CITY, SSNUMBER from SSN`
- Puis sélectionnez le **tOracleSP** et paramétrez ses propriétés dans l'onglet **Basic settings**.

- Comme pour le composant **tOracleInput**, sélectionnez l'option **Repository** dans le champ **Property type** et cochez la case **Use an existing connection**, puis sélectionnez l'entrée adéquate dans la liste **Component List**.
- Le schéma utilisé pour le composant **tOracleSP** diffère légèrement du schéma d'entrée. En effet, une colonne supplémentaire (*SSN_Valid*) est ajoutée au schéma d'entrée. Cette colonne contiendra le statut de validité (*1* ou *0*) généré par la procédure.

tOracleSP_1 (Output)

Column	D...	Key	Type	Nullable	Dat...	L...	P..	D..
ID		☑	float	☐		22	10	
NAME		☐	String	☐		50		
CITY		☐	String	☐		50		
SSNUMBER		☐	String	☐		12		
SSN_Valid		☐	int	☐		2		

- Puis sélectionnez le type d'encodage adéquat dans la liste **Encoding type**.
- Dans le champ **SP Name**, saisissez le nom exact de la procédure stockée (ou de la fonction) telle qu'elle est appelée dans la base de données. Dans cet exemple, le nom de la procédure stockée est *is_ssn*.
- La fonction de base utilisée pour cet exemple est :

```
CREATE OR REPLACE FUNCTION is_ssn(string_in VARCHAR2) RETURN
PLS_INTEGER
IS
-- validating ###-##-#### format
BEGIN
  IF TRANSLATE(string_in, '0123456789A', 'AAAAAAAAAAB') =
     'AAA-AA-AAAA' THEN
    RETURN 1;
  END IF;
  RETURN 0;
END is_ssn;
/
```

- Comme un valeur retournée est attendue dans cet exemple, la procédure agit comme une fonction, ainsi cochez la case **Is function**.
- La seule valeur retournée attendue est basée sur la colonne *ssn_valid*. Ainsi, sélectionnez l'entrée correspondante dans la liste.
- Dans la zone **Parameters**, définissez les paramètres d'entrée et de sortie utilisés dans la procédure. Dans cet exemple, seule la colonne *SSNumber* du schéma est utilisée dans la procédure.
- Cliquez sur le signe **[+]** pour ajouter un nouvelle ligne à la table et sélectionnez la colonne adéquate (*SSNumber*) et son type (*IN*).
- Puis sélectionnez le composant **tLogRow** et cliquez sur **Sync Column** pour récupérer le schéma du composant précédent (**tOracleSP**).

- Cochez la case **Print values in cells of a table** pour faciliter la lecture de la sortie.
- Puis enregistrez le Job et appuyez sur **F6** pour l'exécuter.

```
Starting job OracleSP at 15:14 23/08/2007.
.---+-------+----+-----------+---------.
|               tLogRow_1               |
|=--+-------+----+-----------+--------=|
|ID |NAME   |CITY|SSNUMBER   |SSN_Valid|
|=--+-------+----+-----------+--------=|
|1.0|Jack   |LA  |123-45-6789|1        |
|2.0|Tom    |NYC |123-A5-6789|0        |
|3.0|Bill   |SF  |123=45-6789|0        |
|4.0|Jana   |NYC |236-52-2956|1        |
|5.0|Brandon|SLC |561-52-B267|0        |
'---+-------+----+-----------+---------'

Job OracleSP ended at 15:14 23/08/2007. [exit code=0]
```

Dans la console, vous pouvez lire les résultats en sortie. Toutes les colonnes du schéma d'entrée sont affichées même si elles ne sont pas utilisées comme paramètres dans la procédure stockée.

La colonne finale affiche la valeur retournée attendue, i.e. que le numéro de sécurité social vérifié soit valide ou non.

Consultez également le composant *tParseRecordSet, page 635* si vous voulez analyser un ensemble d'enregistrements d'une table de données ou d'une requête SQL.

tOracleTableList

Propriétés du tOracleTableList

Famille de composant	Databases/Oracle	
Fonction	Le composant **tOracleTableList** effectue une opération d'itération sur toutes les tables d'une base de données, grâce à une connexion Oracle définie.	
Objectif	Le composant **tOracleTableList** permet de lister les noms des tables Oracle grâce aux commandes SELECT et WHERE.	
Basic settings	*Component list*	Sélectionnez le composant de connexion **tOracleConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Where clause for table name selection	Saisissez la commande WHERE permettant d'identifier les tables sur lesquelles effectuer l'opération d'itération.
Utilisation	Il faut utiliser ce composant en association avec les autres composants Oracle, notamment avec le **tOracleConnection**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Itérer une table de base de données et lister le nom des colonnes de la table, page 466*.

tParAccelBulkExec

Propriétés du tParAccelBulkExec

Les composants **tParAccelOutputBulk** et **tParAccelBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tParAccelOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le changement des données dans la base de données.

Famille de composant	Databases/ParAccel	
Fonction	Le composant **tParAccelBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tParAccelBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données Greenplum.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tParAccelConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.

	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit exister pour que l'opération d'Insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
Advanced settings	*Copy mode*	Sélectionnez le mode de Copy que vous souhaitez utiliser. **Basic** : Mode classique, sans optimisation. **Parallel** : Mode permettant d'utiliser plusieurs API internes à ParAccel, afin d'améliorer les performances de chargement.
	Filename	Nom du fichier à traiter et chemin d'accès.
	File Type	Sélectionnez dans la liste le type de fichier.
	Field Layout	Sélectionnez dans la liste la disposition du fichier.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs (si vous chargez un fichier délimité).
	Explicit IDs	L'ID sera assigné directement par la base de données ou est déjà présent dans le fichier à charger.
	Remove Quotes	Cochez cette case afin de retirer les guillemets de votre fichier à charger.
	Max. Errors	Saisissez le nombre maximal d'erreurs avant que votre Job ne s'arrête.
	Date Format	Saisissez le format de date qui sera utilisé.
	Time/Timestamp Format	Saisissez le format de date et heure qui sera utilisé.
	Additional COPY Options	Saisissez l'option spécifique à ParAccel et personnalisée que vous souhaitez utiliser.
	Log file	Saisissez le chemin d'accès à votre fichier de log, ou parcourez votre répertoire.

	Logging level	Sélectionnez le niveau de verbose à retourner.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec un composant **tParAccelOutputBulk**. Ensemble, ils offrent un gain de performance important pour l'alimentation d'une base de données ParAccel.	
Limitation	n/a	

Scénarios associés

Pour plus d'informations relatives au fonctionnement de **tParAccelBulkExec**, consultez les scénarios suivants :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du composant **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du composant **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du composant **tOracleBulkExec**.

tParAccelClose

Propriétés du tParAccelClose

Famille de composant	Databases/ParAccel	
Fonction	Le composant **tParAccelClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données ParAccel.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tParAccelConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tParAccelConnection** et **tParAccelRollback**. Il est généralement utilisé avec un composant **tParAccelConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tParAccelClose**, consultez *tMysqlConnection, page 471*

tParAccelCommit

Propriétés du tParAccelCommit

Ce composant est étroitement lié aux composants **tParAccelCommit** et **tParAccelRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/ParAccel	
Fonction	Le composant **tParAccelCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tParAccelConnection** dans la liste.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tParAccelCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants ParAccel et notamment **tParAccelConnection** et **tParAccelRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tParAccelConnection** et **tParAccelRollback**. Ils sont généralement utilisés avec un composant **tParAccelConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tParAccelCommit**, consultez *tMysqlConnection, page 471*

tParAccelConnection

Propriétés du tParAccelConnection

Ce composant est étroitement lié aux composants **tParAccelCommit** et **tParAccellRollback**. Ils sont généralement utilisés avec un composant **tParAccelConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/ParAccel	
Fonction	Le composant **tParAccelConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*Auto commit*	Cochez cette case pour que, lorsqu'une déclaration est terminée, la méthode commit soit appelée automatiquement sur cette déclaration.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Ce composant est généralement utilisé avec des composants ParAccel, notamment les composants **tParAccelCommit** et **tParAccelRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tParAccelCommit** et **tParAccelRollback**. Ils sont généralement utilisés avec un composant **tParAccelConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tParAccelConnection**, consultez *tMysqlConnection, page 471*

tParAccelInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tParAccelInput

Famille de composant	Databases/ ParAccel	
Fonction	Le composant **tParAccelInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tParAccelInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tParAccelConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.

		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Table name	Nom de la table de base de données.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Use cursor*	Cochez cette case et définissez le nombre de lignes avec lesquelles vous souhaitez travailler en une fois. Cette option permet d'optimiser les performances.
	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimer les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données ParAccel.	

Scénarios associés

Pour un scénario associé, consultez les scénarios du composant **tParAccelInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241*

tParAccelOutput

Propriétés du tParAccelOutput

Famille de composant	Databases/ParAccel	
Fonction	Le composant **tParAccelOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	**Le tParAccelOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tParAccelConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.

	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.
	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du Guide utilisateur de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du Guide utilisateur de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tParAccelOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données ParAccel. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.

Scénarios associés

Pour un scénario associé au composant **tParAccelOutput** consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du **tMysqlOutput**.

tParAccelOutputBulk

Propriétés du tParAccelOutputBulk

Les composants **tParAccelOutputBulk** et **tParAccelBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tParAccelOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le chargement des données dans la base de données.

Famille de composant	Databases/ParAccel	
Fonction	Ecrit un fichier composé de colonnes et basé sur le délimiteur défini et sur les standards ParAccel.	
Objectif	Prépare le fichier à utiliser comme paramètre dans la requête INSERT servant à alimenter une base de données ParAccel.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Row separator*	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Include header	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.

	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec le composant **tParAccelBulkExec**. Ensemble ils offrent un gain de performance important pour l'alimentation d'une base de données ParAccel.	

Scénarios associés

Pour un scénario associé au composant **tParAccelOutputBulk**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du tMysqlOutputBulk.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du tMysqlOutputBulkExec.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du tOracleBulkExec.

tParAccelOutputBulkExec

Propriétés du tParAccelOutputBulkExec

Les composants **tParAccelOutputBulk** et **tParAccelBulkExec**sont généralement utilisés ensemble comme deux parties d'un processus en deux étapes. Dans la première étape, un fichier de sortie est généré. Dans la deuxième étape, ce fichier est utilisé lors de l'opération d'INSERT afin de populer une base de données. Cette exécution en deux étapes est unifiée dans le composant **tParAccelOutputBulkExec**.

Famille de composant	Databases/ParAccel	
Fonction	Le **tParAccelOutputBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tParAccelOutputBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données ParAccel.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'Insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None**: n'effectuer aucune opération de table. **Drop and create the table**: supprimer la table puis en créer une nouvelle. **Create a table**: créer une table qui n'existe pas encore. **Create table if doesn't exist**: créer la table si nécessaire. **Clear a table**: supprimer le contenu de la table.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.

		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	File Name	Nom du fichier à traiter. Voir également :*Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Copy mode*	Sélectionnez le mode de Copy que vous souhaitez utiliser. **Basic** : Mode classique, sans optimisation. **Parallel** : Mode permettant d'utiliser plusieurs API internes à ParAccel, afin d'améliorer les performances de chargement.
	Filename	Nom du fichier à traiter et chemin d'accès.
	File Type	Sélectionnez dans la liste le type de fichier.
	Field Layout	Sélectionnez dans la liste la disposition du fichier.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs (si vous chargez un fichier délimité).
	Explicit IDs	L'ID sera assigné directement par la base de données ou est déjà présent dans le fichier à charger.
	Remove Quotes	Cochez cette case afin de retirer les guillemets de votre fichier à charger.
	Max. Errors	Saisissez le nombre maximal d'erreurs avant que votre Job ne s'arrête.
	Date Format	Saisissez le format de date qui sera utilisé.
	Time/Timestamp Format	Saisissez le format de date et heure qui sera utilisé.
	Additional COPY Options	Saisissez l'option spécifique à ParAccel et personnalisée que vous souhaitez utiliser.
	Log file	Saisissez le chemin d'accès à votre fichier de log, ou parcourez votre répertoire.
	Logging level	Sélectionnez le niveau de verbose à retourner.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est principalement utilisé lorsqu'aucune transformation particulière n'est requise sur les données à charger dans la base de données.	

Scénarios associés

Pour un scénario associé au composant **tParAccelOutputBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tParAccelRollback

Propriétés du tParAccelRollback

Ce composant est étroitement lié aux composants **tParAccelCommit** et **tParAccelConnection**. Ils sont généralement utilisés ensemble lors d'une transaction.

Famille de composant	Databases/ParAccel	
Fonction	Le **tParAccelRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant permet d'éviter le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant **tParAccelConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce component est généralement utilisé avec d'autres composants ParAccel, notamment les composants **tParAccelConnection** et **tParAccelCommit** components.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au composant **tParAccelRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522*.

tParAccelRow

Propriétés du tParAccelRow

Famille de composant	Databases/ParAccel	
Fonction	Le **tParAccelRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tParAccelRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou **Repository**.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tParAccelConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma

	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Table name	Nom de la table de base à lire.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.

	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tParAccelSCD

Le composant **tParAccelSCD** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tParAccelSCD, page 97*.

tParseRecordSet

Propriétés du tParseRecordSet

Ce composant se trouve à la racine de la famille **Databases** de la **Palette** de Talend Open Studio. Le composant **tParseRecordSet** couvre des besoins indirectement liés à l'utilisation des bases de données de tout type.

Famille de composant	Databases	
Fonction	Le composant **tParseRecordSet** analyse un ensemble d'enregistrements d'une table de données ou d'une requête SQL et retourne éventuellement des enregistrements seuls.	
Objectif	Il permet d'analyser un ensemble d'enregistrements d'une table au lieu de les analyser individuellement.	
Basic settings	*Prev. Comp. Column list*	Sélectionnez la colonne contenant l'ensemble d'enregistrements à analyser.
	Schema et *Edit Schema*	Dans une procédure stockée, le schéma est un paramètre d'entrée. Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant uniquement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de Talend Open Studio.
		Repository : Le schéma existe déjà et il est stocké dans le Repository. Ainsi, il est réutilisable. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de Talend Open Studio.
	Attribute table	Paramétrez la valeur positionnelle de chaque colonne de chaque enregistrement d'un ensemble d'enregistrements.
Utilisation	Ce composant est un composant intermédiaire. Il peut être utilisé comme composant de début. Dans ce cas, seuls les paramètres d'entrée sont autorisés.	
Limitation	Ce composant est principalement utilisé avec la fonctionnalité Recordset d'un composant SP.	

Scénario associé

Pour un exemple d'utilisation du **tParseRecordSet**, consultez le *Scénario 2 : Utiliser l'instance PreparedStatement pour faire une requête sur des données, page 527*.

tPostgresPlusBulkExec

Propriétés du tPostgresPlusBulkExec

Les composants **tPostgresPlusOutputBulk** et **tPostgresPlusBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tPostgresPlusOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le changement des données dans la base de données.

Famille de composant	Databases/PostgresPlus	
Fonction	Le composant **PostgresPlusBulkExec** exécute des opérations d'Insert sur les données fournies.	
Objectif	Le **tPostgresPlusBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données DB2.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tPostgresqlConnection** adéquat à partir de la liste **Component List** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.

	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification sur l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None**: n'effectuer aucune opération de table. **Drop and create the table**: supprimer la table puis en créer une nouvelle. **Create a table**: créer une table qui n'existe pas encore. **Create table if doesn't exist**: créer la table si nécessaire. **Clear a table**: supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Filename	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
Advanced settings	*Action*	Sélectionnez l'opération que vous voulez effectuer : **Bulk insert** **Bulk update** Les informations demandées seront différentes en fonction de l'action choisie.
	Field terminated by	Caractère, chaîne ou expression régulière séparant les champs.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes Postgresplus.	

Scénario associé

Pour un scénario associé au composant **tPostgresPlusBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tPostgresPlusClose

Propriétés du tPostgresPlusClose

Famille de composant	Databases/PostgresPlus	
Fonction	Le composant **tPostgresPlusClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données PostgresPlus.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tPostgresPlusConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tPostgresPlusConnection** et **tPostgresPlusRollback**. Il est généralement utilisé avec un composant **tPostgresPlusConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tPostgresPlusClose**, consultez *tMysqlConnection, page 471*

tPostgresPlusCommit

Propriétés du tPostgresPlusCommit

Ce composant est étroitement lié aux composants **tpostgresPlusConnection** et **tPostgresPlusRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/PostgresPlus	
Fonction	Le composant **tPostgresPlusCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tPostgresPlusConnection** dans la liste.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tPostgresPlusCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants PostgresPlus et notamment avec les composants **tPostgresPlusConnection** et **tPostgresPlusRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tPostgresPlusConnection** et **tPostgresPlusRollback**. Ils sont généralement utilisés avec un composant **tPostgresPlusConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tPostgresPlusCommit**, consultez *tMysqlConnection, page 471*.

tPostgresPlusConnection

Propriétés du tPostgresPlusConnection

Ce composant est étroitement lié aux composants **tPostgresPlusCommit** et **tPostgresPlusRollback**. Ils sont généralement utilisés avec un composant **tPostgresPlusConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/PostgresPlus	
Fonction	Le composant **tPostgresPlusConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*Auto commit*	Cochez cette case pour que, lorsqu'une déclaration est terminée, la méthode commit soit appelée automatiquement sur cette déclaration.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Ce composant est généralement utilisé avec des composants PostgresPlus et notamment avec les composants **tPostgresPlusCommit** et **tPostgresPlusRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tPostgresPlusCommit** et **tPostgresPlusRollback**. Ils sont généralement utilisés avec un composant **tPostgresPlusConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tPostgresPlusConnection**, consultez *tMysqlConnection, page 471*

tPostgresPlusInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tPostgresPlusInput

Famille de composant	Databases/ PostgresPlus	
Fonction	Le composant **tPostgresPlusInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tPostgresPlusInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tPostgresPlusConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.

		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Table name	Nom de la table de base de données.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Use cursor*	Cochez cette case et définissez le nombre de lignes avec lesquelles vous souhaitez travailler en une fois. Cette option permet d'optimiser les performances
	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimer les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données PostgresPlus.	

Scénarios associés

Pour un scénario associé, consultez les scénarios du composant **tPostgresPlusInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241*

tPostgresPlusOutput

Propriétés du tPostgresPlusOutput

Famille de composant	Databases/PostgresPlus	
Fonction	Le **tPostgresPlusOutput** écrit, met à jour, effectue les changements ou supprime les entrées dans une base de données.	
Objectif	Le **tPostgresPlusOutput** exécute l'action définie sur la table et/ou sur les données contenues dans la table, à partir du flux entrant du composant précédent dans le Job.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case si vous utilisez composant **tPostgresPlusConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'Insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete)*
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.

Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tPostgresqlOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	Support null in "SQL WHERE" statement	Cochez cette case pour prendre en compte les valeurs Null d'une table de base de données. Assurez-vous que la case Nullable est bien cochée pour les colonnes du schéma correspondantes.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données PostgresPlus. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour des scénarios associés, consultez :

- **tDBOutput** *Scénario : Afficher la base de données en sortie, page 245*

- **tMysqlOutput** *Scénario 1 : Ajouter une colonne et modifier les données, page 493.*

tPostgresPlusOutputBulk

Propriétés du tPostgresPlusOutputBulk

Les composants **tPostgresPlusOutputBulk** et **tPostgresPlusBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tPostgresPlusOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le chargement des données dans la base de données.

Famille de composant	Databases/PostgresPlus	
Fonction	Ecrit un fichier composé de colonnes et basé sur le délimiteur défini et sur les standards Postgresql.	
Objectif	Prépare le fichier à utiliser comme paramètre dans la requête INSERT servant à alimenter une base de données Postgresql.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Row separator*	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Include header	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.

	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec le composant **tPostgresPlusBulkExec**. Ensemble ils offrent un gain de performance important pour l'alimentation d'une base de données PostgresPlus.	

Scénarios associés

Pour un scénario associé au composant **tPostgresPlusOutputBulk**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du tMysqlOutputBulk.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du tMysqlOutputBulkExec.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du tOracleBulkExec.

tPostgresPlusOutputBulkExec

Propriétés du tPostgresplusOutputBulkExec

Les composants **tPostgresPlusOutputBulk** et **tPostgresPlusBulkExec** sont généralement utilisés ensemble comme deux parties d'un processus en deux étapes. Dans la première étape, un fichier de sortie est généré. Dans la deuxième étape, ce fichier est utilisé lors de l'opération d'INSERT afin de populer une base de données. Cette exécution en deux étapes est unifiée dans le composant **tPostgresPlusOutputBulkExec**.

Famille de composant	Databases/Postgresplus	
Fonction	Le **tPostgresPlusOutputBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tPostgresPlusOutputBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données PostgresPlus.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'Insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Action*	Sélectionnez l'opération que vous voulez effectuer : **Bulk insert** **Bulk update** Les informations demandées seront différentes en fonction de l'action choisie.
	File type	Sélectionnez le type de fichiers à traiter.
	Null string	Chaîne de caractères affichée pour signifier que la valeur est nulle.
	Row separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Fields terminated by	Caractère, chaîne ou expression régulière séparant les champs : **None** : aucun caractère de séparation. **Whitespace** : le caractère de séparation est un espace. **EOF (used for loading LOBs from lobfile)** : la séparation est constituée d'un caractère de fin de fichier (End Of File). **Other terminator** : la séparation est constituée d'un caractère autre que ceux précédemment énoncés.
	Escape character	Caractère d'échappement de la ligne.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est principalement utilisé lorsqu'aucune transformation particulière n'est requise sur les données à charger dans la base de données.	

Scénarios associés

Pour un scénario associé au composant **tPostgresPlusOutputBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tPostgresPlusRollback

Propriétés du tPostgresPlusRollback

Ce composant est étroitement lié aux composants **tPostgresPlusCommit** et **tPostgresPlusConnection**. Ils sont généralement utilisés ensemble lors d'une transaction.

Famille de composant	Databases/PostgresPlus	
Fonction	Le **tPostgresPlusRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant permet d'éviter le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant **tPostgresPlusConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce component est généralement utilisé avec d'autres composants PostgresPlus, notamment les composants **tPostgresPlusConnection** et **tPostgresPlusCommit** components.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au composant **tPostgresPlusRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522*.

tPostgresPlusRow

Propriétés du tPostgresPlusRow

Famille de composant	Databases/PostgresPlus	
Fonction	Le **tPostgresPlusRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tPostgresPlusRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou **Repository**.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tPostgresPlusConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma

	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Table name	Nom de la table de base à lire.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.

	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tPostgresPlusSCD

Le composant **tPostgresPlusSCD** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tPostgresPlusSCD, page 99*.

tPostgresPlusSCDELT

Le composant **tPostgresPlusSCDELT** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tPostgresPlusSCDELT, page 101*.

tPostgresqlBulkExec

Propriétés du tPostgresqlBulkExec

Les composants **tPostgresqlOutputBulk** et **tPostgresqlBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tPostgresqlOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le chargement des données dans la base de données.

Famille de composant	Databases/Postgresql	
Fonction	Le composant **tPostgresqlBulkExec** permet d'insérer les données d'une base de données Postgresql.	
Objectif	Le **tPostgresqlBulkExec** est un composant dédié, il permet un gain de performance lors du traitement des données de la base de données Postgresql.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tPostgresqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.

	Database	Nom de la base de données.
	Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit exister pour que l'opération d'Insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
Advanced settings	*Action on data*	Sélectionnez l'opération que vous voulez effectuer : **Bulk insert** **Bulk update** Les informations demandées seront différentes en fonction de l'action choisie.
	Copy the OID for each row	Récupère les identifiants d'objet pour chaque ligne.
	Contains a header line with the names of each column in the file	Spécifiez que la table contient des lignes d'en-tête.
	File type	Sélectionnez le type de fichiers à traiter.
	Null string	Chaîne de caractères affichée pour signifier que la valeur est nulle.
	Fields terminated by	Caractère, chaîne ou expression régulière séparant les champs.
	Escape char	Caractère d'échappement de la ligne.
	Text enclosure	Caractères utilisés pour entourer le texte.
	Use standard_conforming _string ON	Activez la variable

	Force not null for columns	Définissez la nullabilité des colonnes **Force not null** : Cochez la case correspondant à la colonne (**Column**) que vous souhaitez définir comme n'étant pas nulle.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant doit être utilisé en association avec le composant **tPostgresqlOutputBulk**. Ensemble, ils permettent d'obtenir des gains de performances dans une base de données Postgresql.	
Limitation	n/a	

Scénarios associés

Pour d'autres exemples d'utilisation du composant **tPostgresqlBulkExec**, consultez les scénarios suivants :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulk**,
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**,
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tPostgresqlClose

Propriétés du tPostgresqlClose

Famille de composant	Databases/Postgresql	
Fonction	Le composant **tPostgresqlClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données Postgresql.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tPostgresqlConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tPostgresqlConnection** et **tPostgresqlRollback**. Il est généralement utilisé avec un composant **tPostgresqlConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tPostgresqlClose**, consultez *tMysqlConnection, page 471*

tPostgresqlCommit

Propriétés du tPostgresqlCommit

Ce composant est étroitement lié aux composants **tPostgresqlConnection** et **tPostgresqlRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Postgresql	
Fonction	Le composant **tPostgresqlCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	Sélectionnez le composant **tPostgresqlConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tPostgresqlCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants Postgresql, notamment les composants **tPostgresqlConnection** et **tPostgresqlRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tPostgresqlConnection** et **tPostgresqlRollback**. Ils sont généralement utilisés avec un composant **tPostgresqlConnection** car celui-ci permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tPostgresqlCommit**, consultez *tMysqlConnection, page 471*.

tPostgresqlConnection

Propriétés du tPostgresqlConnection

Ce composant est étroitement lié aux composants **tPostgresqlCommit** et **tPostgresqlRollback**. Ils sont généralement utilisés avec un composant **tPostgresqlConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/Postgresql	
Fonction	Le composant **tPostgresqlConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom du schéma
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*Auto commit*	Cochez cette case pour que, lorsqu'une déclaration est terminée, la méthode commit soit appelée automatiquement sur cette déclaration.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Ce composant est généralement utilisé avec des composants Oracle, notamment les composants **tPostgresqlCommit** et **tPostgresqlRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tPostgresqlCommit** et **tPostgresqlRollback**. Ils sont généralement utilisés avec un composant **tPostgresqlConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tPostgresqlConnection**, consultez *tMysqlConnection, page 471*

tPostgresqlInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tPostgresqlInput

Famille de composant	Databases/ Postgresql	
Fonction	Le composant **tPostgresqlInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tPostgresqlInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tPostgresqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.

		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à lire.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Use cursor*	Cochez cette case et définissez le nombre de lignes avec lesquelles vous souhaitez travailler en une fois. Cette option permet d'optimiser les performances.
	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimer les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Postgresql.	

Scénarios associés

Pour un scénario associé, consultez les scénarios du composant **tPostgresqlInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241*

tPostgresqlOutput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tPostgresqlOutput

Famille de composant	Databases/Postgresql	
Fonction	Le composant **tPostgresqlOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	**Le tPostgresqlOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tPostgresqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de la base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).* La fonction de schéma dynamique peut être utilisée dans différents modes : **Insert**, **Update**, **Insert or Update**, **Update or insert**, **Delete**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.

		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tPostgresqlOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	Support null in "SQL WHERE" statement	Cochez cette case pour prendre en compte les valeurs Null d'une table de base de données. Assurez-vous que la case Nullable est bien cochée pour les colonnes du schéma correspondantes.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données Postgresql. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.

Scénarios associés

Pour un scénario associé au composant **tPostgresqlOutput** consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du **tMysqlOutput**.
- le *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507* du composant **tMysqlOutput**.

tPostgresqlOutputBulk

Propriétés du tPostgresqlOutputBulk

Les composants **tPostgresqlOutputBulk** et **tPostgresqlBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tPostgresqlOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le chargement des données dans la base de données.

Famille de composant	Databases/Postgresql	
Fonction	Ecrit un fichier composé de colonnes et basé sur le délimiteur défini et sur les standards Postgresql.	
Objectif	Prépare le fichier à utiliser comme paramètre dans la requête INSERT servant à alimenter une base de données Postgresql.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Row separator*	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Include header	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.

	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec le composant **tPostgresqlBulkExec**. Ensemble ils offrent un gain de performance important pour l'alimentation d'une base de données Postgresql.	

Scénarios associés

Pour un scénario associé au composant **tPostgresqlOutputBulk**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du tMysqlOutputBulk.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du tMysqlOutputBulkExec.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du tOracleBulkExec.

tPostgresqlOutputBulkExec

Propriétés du tPostgresqlOutputBulkExec

Les composants **tPostgresqlOutputBulk** et **tPostgresqlBulkExec** sont généralement utilisés ensemble comme deux parties d'un processus en deux étapes. Dans la première étape, un fichier de sortie est généré. Dans la deuxième étape, ce fichier est utilisé lors de l'opération d'INSERT afin de populer une base de données. Cette exécution en deux étapes est unifiée dans le composant **tPostgresqlOutputBulkExec**.

Famille de composant	Databases/Postgresql	
Fonction	Le **tPostgresqlOutputBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tPostgresqlOutputBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données Postgresql.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'Insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None**: n'effectuer aucune opération de table. **Drop and create the table**: supprimer la table puis en créer une nouvelle. **Create a table**: créer une table qui n'existe pas encore. **Create table if doesn't exist**: créer la table si nécessaire. **Drop a table if exists and create**: supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table**: supprimer le contenu de la table.

	File Name	Nom du fichier à traiter. Voir également :*Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Action on data*	Sélectionnez l'opération que vous voulez effectuer : **Bulk insert** **Bulk update** Les informations demandées seront différentes en fonction de l'action choisie.
	Copy the OID for each row	Récupère les identifiants d'objet pour chaque ligne.
	Contains a header line with the names of each column in the file	Spécifiez que la table contient des lignes d'en-tête.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	File type	Sélectionnez le type de fichiers à traiter.
	Null string	Chaîne de caractères affichée pour signifier que la valeur est nulle.
	Row separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Fields terminated by	Caractère, chaîne ou expression régulière séparant les champs.
	Escape char	Caractère d'échappement de la ligne.
	Text enclosure	Caractères utilisés pour entourer le texte.
	Activate standard_conforming _string	Activez le paramètre standard_conforming_string pour traiter les backslashs comme une chaîne ordinaire et non pas comme un caractère d'échappement.
	Force not null for columns	Définissez la nullabilité des colonnes **Force not null** : Cochez la case correspondant à la colonne (**Column**) que vous souhaitez définir comme n'étant pas nulle.

	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est principalement utilisé lorsqu'aucune transformation particulière n'est requise sur les données à charger dans la base de données.	

Scénarios associés

Pour un scénario associé au cmposant **tPostgresqlOutputBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tPostgresqlRollback

Propriétés du tPostgresqlRollback

Ce composant est étroitement lié aux composants **tPostgresqlCommit** et **tPostgresqlConnection**. Ils sont généralement utilisés ensemble lors d'une transaction.

Famille de composant	Databases/Postgresql	
Fonction	Le **tPostgresqlRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant **tPostgresqlConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec d'autres composants Oracle, notamment les composants **tPostgresqlConnection** et **tPostgresqlCommit**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tPostgresqlConnection** et **tPostgresqlCommit**. Ils sont généralement utilisés avec un composant **tPostgresqlConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tPostgresqlRollback**, consultez *tMysqlRollback, page 522*

tPostgresqlRow

Propriétés du tPostgresqlRow

Famille de composant	Databases/Postgresql	
Fonction	Le **tPostgresqlRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tPostgresqlRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou **Repository**.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tPostgresqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Schema	Nom exact du schéma

	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à lire.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.

	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénarios associés

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tPostgresqlSCD

Le composant **tPostgresqlSCD** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tPostgresqlSCD, page 104*.

tPostgresqlSCDELT

Le composant **tPostgresqlSCDELT** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tPostgresqlSCDELT, page 106*.

tSASInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Assurez-vous d'installer au préalable les modules sas.core.jar, sas.intrnet.javatools.jar et sas.svc.connection.jar sous le chemin **lib > java** *dans votre build* ***Talend Open Studio*** *pour pouvoir utiliser toutes les fonctionnalités des composants SAS. Vous pouvez vérifier que ces modules ont bien été installés à partir de la vue* **Modules** *de votre studio.*

Propriétés du tSASInput

Famille de composant	Databases/SAS	
Fonction	Le composant **tSASInput lit une base de données et en extrait des champs à l'aide de requêtes.**	
Objectif	Le **tSASInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio** .
	Host name	Nom ou adresse IP du serveur SAS.
	Port	Port d'écoute à ouvrir.
	Librefs	Renseignez le nom du répertoire physique dans lequel est contenue la table à lire, suivi de son chemin d'accès. On aura par exemple : `"TpSas 'C:/SAS/TpSas'"`
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table Name	Renseignez le nom de la table à lire, préfixé par le nom du répertoire physique dans lequel elle est contenue. On aura par exemple : `"TpSas.Customers"`.
	Query type	La requête peut être construite manuellement (Built-in) pour un Job particulier ou stockée dans le **Repository** s'il s'agit d'une requête fréquemment utilisée, pour en faciliter l'utilisation. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
	Query	Saisissez votre requête, si elle n'est pas stockée dans le **Repository**, en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données utilisant une connexion SAS.	

Scénarios associés

Pour un scénario associé, consultez les scénarios du composant **tDBInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239.*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241.*

Consultez également le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du composant **tContextLoad**.

tSASOutput

Assurez-vous d'installer au préalable les modules sas.core.jar, sas.intrnet.javatools.jar et sas.svc.connection.jar sous le chemin **lib > java** *dans votre build* ***Talend Open Studio*** *pour pouvoir utiliser toutes les fonctionnalités des composants SAS. Vous pouvez vérifier que ces modules ont bien été installés à partir de la vue* **Modules** *de votre studio.*

Propriétés du tSASOutput

Famille de composant	Databases/SAS	
Fonction	Le composant **tSASOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tSASOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Use an existing connection*	Cochez cette case et sélectionnez le composant **tSASConnection** adéquat à partir de la liste **Component** pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	SAS URL	Entrez le chemin d'accès à la base de données.
	Driver JAR	Sélectionnez le driver JAR à partir de la liste déroulante ou cliquez sur les trois points **[...]** pour ajouter un nouveau JAR à la liste.
	Class Name	Dans le champ **Class name, renseignez le nom de la classe pour qu'elle soit indiquée dans le driver.**
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.

	Table	Nom de la table à lire.
	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*
	Clear data in table	Cochez cette case pour que les données soient supprimées de la table sélectionnée avant toute action.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.

Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tSASOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données utilisant une connexion SAS. Il permet de faire des actions sur une table ou les données d'une table d'une base de données SAS. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour un scénario associé au **tSASOutput**, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMysqlOutput**.

tSQLiteClose

Propriétés du tSQLiteClose

Famille de composant	Databases/SQLite	
Fonction	Le composant **tSQLiteClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données SQLite.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tSQLiteConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tSQLiteConnection** et **tSQLiteRollback**. Il est généralement utilisé avec un composant **tSQLiteConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tSQLiteClose**, consultez *tMysqlConnection, page 471*

tSQLiteCommit

Propriétés du tSQLiteCommit

Ce composant est étroitement lié aux composants **tSQLiteConnection** et **tSQLiteRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/SQLite	
Fonction	Le composant **tSQLiteCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	Sélectionnez le composant **tSQLiteConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tSQLiteCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants SQLite, notamment les composants **tSQLiteConnection** et **tSQLiteRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tSQLiteConnection** et **tSQLiteRollback**. Ils sont généralement utilisés avec un composant **tSQLiteConnection** car celui-ci permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tSQLiteCommit**, consultez *tMysqlConnection, page 471*.

tSQLiteConnection

Propriétés du tSQLiteConnection

Ce composant est étroitement lié aux composants **tSQLiteCommit** et **tSQLiteRollback**. Ils sont généralemet utilisés avec un composant **tSQLiteConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/SQLite	
Fonction	Le composant **tSQLiteConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Database	Nom de la base de données.
	Table Schema	Nom du schéma.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*Auto commit*	Cochez cette case pour que, lorsqu'une déclaration est terminée, la méthode commit soit appelée automatiquement sur cette déclaration.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Ce composant est généralement utilisé avec des composants SQLite, notamment les composants **tSQLiteCommit** et **tSQLiteRollback**.	
Limitation	n/a	

Scénarios associés

Ce composant est étroitement lié aux composants **tSQLiteCommit** et **tSQLiteRollback**. Ils sont généralement utilisés avec un composant **tSQLiteConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tSQLiteConnection**, consultez *tMysqlConnection, page 471*

tSQLiteInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tSQLiteInput

Famille de composant	Databases/SQLite	
Fonction	Le composant **tSQLiteInput** lit une base de données et en extrait des champs à l'aide de requêtes. Un petit moteur de base de données SQLite lui est intégré, il n'a donc pas besoin d'être connecté à un serveur de base de données.	
Objectif	**Le tSQLiteInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tSQLiteConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.

		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio** .
	Database	Chemin d'accès au fichier de la base de données SQLite.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Query type	La requête peut être construite manuellement (Built-in) pour un Job particulier ou stockée dans le **Repository** s'il s'agit d'une requête fréquemment utilisée, pour en faciliter l'utilisation.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Trim all the String/Char columns*	Supprimer les espaces en début et en fin de champ dans les colonnes sélectionnées.
	Trim column	Cochez cette case pour collecter les données de log au niveau du composant.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant est tout à fait autonome du fait de son moteur de base de données SQLite intégré. Il s'agit d'un composant de début pouvant initier un traitement de flux de données.

Scénario : Filtrer des données SQlite

Le scénario suivant décrit un Job assez simple qui permet d'appliquer une requête donnée utilisant un filtre pour extraire des lignes à partir d'une base de données SQLite source et alimenter une table SQLite en sortie.

- Cliquez-déposez à partir de la **Palette** les composants **tSQLiteInput** et **tSQLiteOutput**.
- Reliez le composant d'entrée et de sortie à l'aide d'un lien de type **row Main**.
- Dans l'onglet **Basic settings** du composant **tSQLiteInput**, entrez le chemin d'accès ou parcourez vos dossiers jusqu'au fichier d'entrée à la base de donnée SQLite.

	id	version	download_date	ip	type	type_os
1	20027	TOS-Win32-200610	13/11/2006	1191947907	1	101
2	20028	TOS-Win32-200610	13/11/2006	1195650472	1	102
3	20030	TOS-Win32-200610	13/11/2006	3375565745	1	103
4	20031	TOS-Win32-200610	13/11/2006	1195650472	1	104
5	20032	TOS-Win32-200610	13/11/2006	1195650472	1	105
6	20033	TOS-Win32-200610	13/11/2006	1104872453	1	106
7	20034	TOS-Win32-200610	13/11/2006	1104872453	1	107
8	20036	TOS-Win32-200610	13/11/2006	1190898057	1	108
9	20037	TOS-Win32-200610	13/11/2006	1190898057	1	109
10	20038	TOS-Win32-200610	13/11/2006	1348977142	1	110
11	20040	TOS-Win32-200610	13/11/2006	3581349521	1	11
12	20041	TOS-Win32-200610	13/11/2006	1190898057	1	12
13	20043	TOS-Win32-200610	13/11/2006	1196485544	1	13
14	20044	TOS-Win32-200610	13/11/2006	1066743463	1	14
15	20045	TOS-Win32-200610	13/11/2006	1196485544	1	15
16	20046	TOS-Win32-200610	13/11/2006	3624217794	1	16

- Le fichier contient des centaines de lignes et comprend une colonne **ip** servant de base à la requête.
- Dans l'onglet **Basic settings** du composant **tSQLiteInput**, modifiez le schéma afin qu'il corresponde à la structure de la table.

- Dans le champ **Query**, renseignez la requête choisie par rapport à la colonne *ip*.
- Sélectionnez les paramètres d'encodage correspondants.
- Pour le composant **tSQLiteOutput**, dans le champ **Database** de l'onglet **Basic settings**, sélectionnez le chemin d'accès à la base de données de sortie.

- Dans le champ **Table**, renseignez le nom de la table à alimenter avec les données sélectionnées.
- A partir des listes déroulantes **Action on table** et **Action on Data**, sélectionnez les actions à effectuer sur la table et sur les données. Dans le cas présent, les actions sont, respectivement, *Drop and create* et *Insert*.
- Synchronisez le schéma avec le schéma d'entrée.
- Sélectionnez l'encodage puis définissez la limite de commit.
- Enregistrez le Job puis exécutez-le en appuyant sur **F6**.

	id	version	download_date	ip	type	type_os
1	20028	TOS-Win32-200610	13/11/2006	1195650472	1	102
2	20031	TOS-Win32-200610	13/11/2006	1195650472	1	104
3	20032	TOS-Win32-200610	13/11/2006	1195650472	1	105

Les données sélectionnées sont renvoyées vers le fichier SQLite défini.

tSQLiteOutput

Propriétés du tSQLiteOutput

Famille de composant	Databases/SQLite	
Fonction	Le composant **tSQLiteOutput** écrit, met à jour, modifie ou supprime les données d'une base de données. Un petit moteur de base de données SQLite lui est intégré, il n'a donc pas besoin d'être connecté à un serveur de base de données.	
Objectif	Le **tSQLiteOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tSQLiteConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.

		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio** .
	Database	Chemin d'accès au fichier de la base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tSQLiteOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant doit absolument être relié à un composant d'entrée (Input). Il permet de faire des actions sur une table ou les données d'une table d'une base de données SQLite. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.

Scénario associé

Pour un scénario associé au composant **tSQLiteOutput**, consultez *tSQLiteInput, page 697*.

tSQLiteRollback

Propriétés du tSQLiteRollback

Ce composant est étroitement lié aux composants **tSQLiteCommit** et **tSQLiteConnection**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/SQLite	
Fonction	Le **tSQLiteRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant de connexion **tSQLiteConnection** dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Il faut utiliser ce composant en association avec des composants SQLite, notamment avec le **tSQLiteConnection** et le **tSQLiteCommit**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au **tSQLiteRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522* du composant **tMySQLRollback**.

tSQLiteRow

Propriétés du tSQLiteRow

Famille de composant	Databases/SQLite	
Fonction	Le **tSQLiteRow** exécute une requête définie dans une base de données spécifiée et utilise les paramètres de la colonne.	
Objectif	Une requête prédéfinie utilise le flux d'entrée pour remplacer les paramètres substituables par les valeurs définies pour chaque paramètre. Ce composant est très utile quand il s'agit de faire des mises à jour.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tPostgresqlConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Database	Chemin d'accès au fichier de la base de données SQLite.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.

		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à lire.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes avant le commit.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénario : Mettre à jour des lignes SQLite

Le scénario suivant décrit un Job qui permet de mettre à jour un fichier de base de donnée SQLite en utilisant une requête prédéfinie et un fichier délimité.

- Cliquez-déposez les composants **tFileInputDelimited** et **tSQLiteRow**.
- Dans l'onglet **Basic settings** du composant **tFileInputDelimited**, parcourez les dossiers et sélectionnez le fichier d'entrée à utiliser pour mettre à jour les lignes dans la base de données.

- Il n'y a ni en-tête (**Header**) ni pied de page (**Footer**). Le séparateur de lignes (**Row separator**) est un retour chariot et le séparateur de champs (**Field separator**) un point-virgule.
- Modifiez le schéma s'il n'est pas stocké dans le **Repository**.

Column	Key	Type	Nullable	Length	Precision	Comment
id	☐	int	☑	6		
version	☐	String	☑	40		
download_date	☐	String	☑	20		
ip	☐	String	☑	20		
type	☐	int	☑	1		
type_os	☐	int	☑	3		

- Assurez-vous que la longueur et le type définissent correctement les colonnes.

Dans l'espace de modélisation graphique, double-cliquez sur le composant **tSQLiteRow** pour afficher sa vue **Component**.

- Dans l'onglet **Basic settings** du composant **tSQLiteRow**, renseignez le champ **Database** avec le chemin d'accès au fichier à mettre à jour.

- Le schéma est en mode read-only puisqu'il doit correspondre au schéma d'entrée.
- Saisissez la requête ou récupérez-la à partir du **Repository**. Dans le cas présent, la colonne *type_os* a été mise à jour en fonction de la valeur *id* définie dans le flux entrant. La requête se présente comme suit : `"Update download set type_os=? where id=?"`
- Dans l'onglet **Advanced Settings**, cochez la case **Use PreparedStatement** pour afficher la table des paramètres substituables.

- Dans la table des paramètres d'entrée, ajoutez autant de lignes qu'il y a de paramètres substituables. Dans cet exemple, définissez *type_os* et *id*.
- Renseignez le champ **Commit every**.
- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

La table *download* de la base de données SQLite est ainsi mise à jour avec le nouveau code *type_os*, selon le fichier délimité d'entrée.

tSybaseBulkExec

Propriétés du tSybaseBulkExec

Les composants **tSybaseOutputBulk** et **tSybaseBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tSybaseOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le chargement des données dans la base de données.

Famille de composant	Databases	
Fonction	Le composant **tSybaseBulkExec** permet d'insérer les données d'une base de données Sybase.	
Objectif	Le **tSybaseBulkExec** est un composant dédié, il permet un gain de performance lors du traitement des données de la base de données Sybase.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tSybaseConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Server	Adresse IP du serveur de base de données.

	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Server	Adresse IP du serveur de base de données pour la connexion de l'utilitaire Bcp.
	Bcp Utility	Nom de l'utilitaire bcp à utiliser pour copier les données sur le serveur Sybase.
	Batch size	Nombre de lignes dans chaque lot de traitement de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'Insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération sur la table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Action on data*	Vous pouvez effectuer les actions suivantes sur la table définie : **Bulk Insert** : ajoute des entrées multiples à la table. Le Job s'arrête lorsque des doublons sont trouvés. **Bulk Update** : effectue des changements simultanés aux multiples entrées.
	Field Terminator	Caractère, chaîne ou expression régulière séparant les champs.

	Row Terminator	Chaîne (ex : “\n” sous Unix) séparant les lignes.
	Head row	Nombre de lignes d’en-tête à ignorer en début de fichier.
	Encoding	Sélectionnez l’encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	Output	Sélectionnez la sortie à utiliser pour récupérer la sortie standard de la base Sybase : **to console** : vers la console. **to global variable** : vers la variable globale.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du Job , ainsi qu’au niveau du composant.
Utilisation	Ce composant est généralement utilisé lorsque les données à charger dans la base n’exigent aucune transformation particulière.	
Limitation	Contrairement au composant Oracle dédié, ce composant Sybase ne permet aucune action directe sur les données.	

Scénarios associés

Pour un scénario associé au composant **tSybaseBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tSybaseClose

Propriétés du tSybaseClose

Famille de composant	Databases/Sybase	
Fonction	Le composant **tSybaseClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données Sybase.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tSybaseConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tSybaseConnection** et **tSybaseRollback**. Il est généralement utilisé avec un composant **tSybaseConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tSybaseClose**, consultez *tMysqlConnection, page 471*

tSybaseCommit

Propriétés du tSybaseCommit

Ce composant est étroitement lié aux composants **tSybaseConnection** et **tSybaseRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Sybase	
Fonction	Le composant **tSybaseCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	Sélectionnez le composant **tSybaseCommit** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close Connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tSybaseCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants Sybase, notamment les composants **tSybaseConnection** et **tSybaseRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tSybaseConnection** et **tSybaseRollback**. Ils sont généralement utilisés avec un composant **tSybaseConnection** car celui-ci permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tSybaseCommit**, consultez le *Scénario : Insérer des données dans des tables mère/fille, page 471*.

tSybaseConnection

Propriétés du tSybaseConnection

Ce composant est étroitement lié aux composants **tSybaseCommit** et **tSybaseRollback**. Ils sont généralement utilisés avec un composant **tSybaseConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/Sybase	
Fonction	Le composant **tSybaseConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier une fois qu'il est validé en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Table Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Utilisation	Ce composant est généralement utilisé avec des composants Oracle, notamment les composants **tSybaseCommit** et **tSybaseRollback**.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé au composant **tSybaseConnection**, consultez le *Scénario : Insérer des données dans des tables mère/fille, page 471*.

tSybaseInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tSybaseInput

Famille de composant	Databases/Sybase	
Fonction	Le composant **tSybaseInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tSybaseInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez la section *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio** .

	Use an existing connection	Cochez cette case et sélectionnez le composant **tSybaseConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Server	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Sybase Schema	Nom exact du schéma Sybase.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.

		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à lire.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Trim all the String/Char columns*	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimer les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Sybase.	

Scénario associé

Pour un scénario associé, consultez les scénarios du composant **tDBInput** :

- le *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239*
- le *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241*

Consultez également le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du **tContextLoad**..

tSybaseIQBulkExec

Propriétés du tSybaseIQBulkExec

Famille de composant	Databases/SybaseIQ	
Fonction	Le composant **tSybaseIQBulkExec** permet de charger un fichier bulk dans une base de données Sybase IQ.	
Objectif	Le **tSybaseIQBulkExec** est un composant dédié, il permet un gain de performance lors du traitement des données de la base de données Sybase IQ.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tSybaseConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	DB Version	Sélectionnez la version de Sybase que vous utilisez.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.

	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'Insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération sur la table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Local filename	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Lines terminated by*	Caractère, chaîne ou expression régulière séparant les lignes.
	Field Terminated by	Caractère, chaîne ou expression régulière séparant les champs.
	Use enclosed quotes	Cochez cette case pour utiliser des caractères pour entourer les données.
	Use fixed length	Cochez cette case pour déterminer une longueur fixe pour les lignes de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données Sybase IQ.	
Limitation	Contrairement au composant Oracle dédié bulk, il n'est possible d'effectuer aucune action sur les données avec ce composant dédié Sybase IQ.	

Scénario associé

Pour un scénario associé au **tSybaseIQBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tSybaseIQOutputBulkExec

Propriétés du tSybaseIQOutputBulkExec

Famille de composant	Databases/SybaseIQ	
Fonction	Le **tSybaseIQOutputBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tSybaseIQOutputBulkExec** est un composant dédié, il permet un gain de performance lors du traitement des données de la base de données Sybase IQ.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tSybaseConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.

	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'Insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération sur la table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Append the file	Cochez cette case pour ajouter des lignes à la fin des enregistrements.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Fields terminated by*	Caractère, chaîne ou expression régulière séparant les champs.
	Lines terminated by	Caractère, chaîne ou expression régulière séparant les lignes.
	Use enclose quotes	Cochez cette case pour utiliser des caractères pour entourer les données.
	Include Head	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Usage	Ce composant est principalement utilisé lorsqu'aucune transformation particulière n'est requise sur les données à charger dans la base de données.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé au composant **tSybaseIQOutputBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tSybaseOutput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

tSybaseOutput Properties

Famille de composant	Databases/Sybase	
Fonction	Le composant **tSybaseOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	**Le tSybaseOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case et sélectionnez le composant **tSybaseConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Server	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Sybase Schema	Nom exact du schéma Sybase.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'Insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.

	Turn on identity insert	Cochez cette case pour utiliser votre propre séquence sur les valeurs Identity des données insérées (plutôt que de laisser le serveur SQL choisir les valeurs séquentielles).
	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).* La fonction de schéma dynamique peut être utilisée dans différents modes : **Insert**, **Update**, **Insert or Update**, **Update or insert**, **Delete**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.

		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tSybaseOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données Sybase. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.

Scénario associé

Pour un scénario associé au composant **tSybaseOutput**, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du **tMysqlOutput**.
- le *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507* du composant **tMysqlOutput**.

tSybaseOutputBulk

Propriétés du tSybaseOutputBulk

Les composants **tSybaseOutputBulk** et **tSybaseBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tSybaseOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le chargement des données dans la base de données.

Famille de composant	Databases/Sybase	
Fonction	Ecrit un fichier composé de colonnes et basé sur le délimiteur défini et sur les standards Sybase.	
Objectif	Prépare le fichier à utiliser comme paramètre dans la requête INSERT servant à alimenter une base de données Sybase.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Row separator*	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Include header	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.

	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec le composant **tSybaseBulkExec**. Ensemble ils offrent un gain de performance important pour l'alimentation d'une base de données Sybase.	

Scénario associé

Pour un scénario associé au composant **tSybaseOutputBulk**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tSybaseOutputBulkExec

Propriétés du tSybaseOutputBulkExec

Les composants **tSybaseOutputBulk** et **tSybaseBulkExec** sont généralement utilisés ensemble comme deux parties d'un processus en deux étapes. Dans la première étape, un fichier de sortie est généré. Dans la deuxième étape, ce fichier est utilisé lors de l'opération d'INSERT afin de populer une base de données. Cette exécution en deux étapes est unifiée dans le composant **tSybaseOutputBulkExec**.

Famille de composant	Databases/Sybase	
Fonction	Le **tSybaseOutputBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tSybaseOutputBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données Sybase.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tSybaseConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Server	Adresse IP du serveur de base de données.

	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Bcp Utility	Nom de l'utilitaire bcp à utiliser pour copier les données sur le serveur Sybase.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'Insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Action on data*	Vous pouvez effectuer les actions suivantes sur la table définie : **Bulk Insert** : ajoute des entrées multiples à la table. Le Job s'arrête lorsque des doublons sont trouvés. **Bulk Update** : effectue des changements simultanés aux multiples entrées.
	Field Terminator	Caractère, chaîne ou expression régulière séparant les champs.
	DB Row Terminator	Chaîne (ex : "\n" sous Unix) séparant les lignes dans la base de données.

	First row N0 . Of File	Nombre de lignes d'en-tête à ignorer en début de fichier.
	FILE Row Terminator	Caractère, chaîne ou expression régulière séparant les lignes dans le fichier.
	Include Head	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	Output	**to console** : l'information est chargée. **to global variable** : valeurs retournées à partir de fichiers de log.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est principalement utilisé lorsqu'aucune transformation particulière n'est requise sur les données à charger dans la base de données.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé au composant **tSybaseOutputBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du **tOracleBulkExec**.

tSybaseRollback

Propriétés du tSybaseRollback

Ce composant est étroitement lié aux composants **tSybaseCommit** et **tSybaseConnection**. Ils sont généralement utilisés ensemble lors d'une transaction.

Famille de composant	Databases/Sybase	
Fonction	Le **tSybaseRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant **tSybaseConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec d'autres composants Oracle, notamment les composants **tSybaseConnection** et **tSybaseCommit**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au composant **tSybaseRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522*.

tSybaseRow

Propriétés du tSybaseRow

Famille de composant	Databases/Sybase	
Fonction	Le **tSybaseRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tSybaseRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou **Repository**.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tSybaseConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Server	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Sybase schema	Nom du schéma Sybase.

	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table name	Nom de la table à lire.
	Turn on identity insert	Cochez cette case pour utiliser votre propre séquence sur les valeurs Identity des données insérées (plutôt que de laisser le serveur SQL choisir les valeurs séquentielles).
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Propagate QUERY's recordset	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.

	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des “?” dans l’instruction SQL définie dans le champ **Query** de l’onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l’instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l’écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d’exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénario associé

Pour un scénario associé au composant **tSybaseRow**, consultez :

- le *Scénario : Réinitialiser l’auto-incrémentation d’une base de données, page 249* du **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du **tMySQLRow**.

tSybaseSCD

Le composant tSybaseSCD appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tSybaseSCD, page 119*.

tSybaseSCDELT

Le composant **tSybaseSCDELT** appartient à deux familles différentes : Business Intelligence et Databases. Pour plus d'informations, consultez *tSybaseSCDELT, page 121*.

tSybaseSP

Propriétés du tSybaseSP

Famille de composant	Databases/Sybase	
Fonction	Le composant **tSybaseSP** appelle une procédure stockée de base de données.	
Objectif	Ce composant permet de centraliser des requêtes multiples ou complexes dans une base de données et de les appeler plus facilement.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier des propriétés du composant. Les champs suivants sont alors pré-remplis.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tSybaseConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	SP Name	Saisissez le nom exact de la Procédure Stockée (SP).
	Is Function / Return result in	Cochez cette case, si une seule valeur doit être retournée. Sélectionnez dans la liste la colonne du schéma sur laquelle est basée la valeur à obtenir.
	Timeout Interval	Durée maximale d'attente des resultats de la procédure stockée.
	Parameters	Cliquez sur le bouton (+) et sélectionnez dans le champ **Schema Columns** les différentes colonnes nécessaires à la procédure. Notez que le schéma de la SP peut contenir plus de colonnes qu'il n'y a de paramètres utilisés dans la procédure. Sélectionnez le **Type** de paramètre : **IN** : paramètre d'entrée (Input) **OUT** : paramètre de sortie (Output)/valeur retournée **IN OUT** : les paramètres d'entrée doivent être retournées sous forme de valeur, même après modifications via la procédure (fonction). **RECORDSET** : les paramètres d'entrée doivent être retournées sous forme d'ensemble de valeurs, au lieu d'une valeur unique. Consultez le composant *tParseRecordSet* si vous voulez analyser un ensemble d'enregistrements d'une table de données ou d'une requête SQL.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Use Multiple SELECT Procedure	Cochez cette case pour utiliser des procéures contenant plusieurs instructions SELECT.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est un composant intermédiaire. Il peut être utilisé comme composant de début. Dans ce cas, seuls les paramètres d'entrée sont autorisés.	
Limitation	La syntaxe de la Procédure Stockée doit correspondre à celle de la base de données.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Trouver le libellé State à l'aide d'une procédure stockée, page 537* du composant **tMysqlSP**.

Consultez également le composant *tParseRecordSet, page 635* si vous voulez analyser un ensemble d'enregistrements d'une table de données ou d'une requête SQL.

tTeradataClose

Propriétés du tTeradataClose

Famille de composant	Databases/Teradata	
Fonction	Le composant **tTeradataClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données Teradata.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tTeradataConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tTeradataConnection** et **tTeradataRollback**. Il est généralement utilisé avec un composant **tTeradataConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tTeradataClose**, consultez *tMysqlConnection, page 471*

tTeradataCommit

Propriétés du tTeradataCommit

Ce composant est étroitement lié aux composants **tTeradataConnection** et **tTeradataRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Teradata	
Fonction	Le composant **tTeradataCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	Sélectionnez le composant **tTeradataConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tTeradataCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants Postgresql, notamment les composants **tTeradataConnection** et **tTeradataRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tTeradataConnection** et **tTeradataRollback**. Ils sont généralement utilisés avec un composant **tTeradataConnection** car celui-ci permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tTeradataCommit**, consultez *tMysqlConnection, page 471*.

tTeradataConnection

Propriétés du tTeradataConnection

Ce composant est étroitement lié aux composants **tTeradataCommit** et **tTeradataRollback**. Ils sont généralemet utilisés avec un composant **tTeradataConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/Teradata	
Fonction	Le composant **tTeradataConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Additional JDBC parameters	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données, afin de supporter des caractères spécifiques. Exemple : CHARSET=KANJISIS_OS pour supporter les caractères japonais.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*Auto Commit*	Cochez cette case pour que, lorsqu'une déclaration est terminée, la méthode commit soit appelée automatiquement sur cette déclaration.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants Teradata, notamment les composants **tTeradataCommit** et **tTeradataRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tTeradataCommit** et **tTeradataRollback**. Ils sont généralement utilisés avec un composant **tTeradataConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tTeradataConnection**, consultez *tMysqlConnection, page 471*

tTeradataFastExport

Propriétés du tTeradataFastExport

Famille de composant	Databases/Teradata	
Fonction	Le composant **tTeradataFastExport** exporte très rapidement des lots de données volumineux d'une table ou d'une vue Teradata.	
Objectif	Le **tTeradataFastExport** exporte des lots de données d'une table Teradata vers un système client ou vers une base de données plus petite.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Execution platform	Sélectionnez le type de système d'exploitation que vous utilisez.
	Database name	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Script generated folder	Parcourez votre répertoire et sélectionnez la destination du fichier qui sera créé.
	Exported file	Nom et chemin d'accès au fichier qui sera créé.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Error file	Parcourez votre répertoire et sélectionnez la destination du fichier dans lesquels seront enregistrés les messages d'erreur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Teradata.

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tTeradataFastLoad

Propriétés du tTeradataFastLoad

Famille de composant	Databases/Teradata	
Fonction	Le composant **tTeradataFastLoad** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tTeradataFastLoad** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Execute Batch every	Nombre de lignes par lot à charger.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Teradata.	

Scénario associé

Aucun scénario n’est disponible pour ce composant.

tTeradataFastLoadUtility

Propriétés du tTeradataFastLoadUtility

Famille de composant	Databases/Teradata	
Fonction	Le composant **tTeradataFastLoadUtility** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tTeradataFastLoadUtility** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (**Main row**).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Execution platform	Sélectionnez le type de système d'exploitation que vous utilisez.
	Database name	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Script generated folder	Parcourez votre répertoire et sélectionnez la destination du fichier qui sera créé.
	Load file	Entrez le nom du fichier à partir duquel vous souhaitez charger les données.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Error file	Parcourez votre répertoire et sélectionnez la destination du fichier dans lesquels seront enregistrés les messages d'erreur.

Advanced settings	*Define character set*	Spécifiez l'encodage des caractères dont vous avez besoin pour utiliser votre système.
	Check point	Saisissez la valeur du point de validation.
	Error files	Saisissez le nom du fichier dans lequel les messages d'erreur sont stockés. Par défaut, le code saisi est `ERRORFILES table_ERR1`, ce qui signifie que les deux tables *table_ERR1* et *table_ERR2* sont utilisées pour enregistrer les messages d'erreur.
	Return fastload error	Cochez cette case pour spécifier le code de retour à partir duquel retourner une erreur dans la console.
	ERRLIMIT	Saisissez le nombre limite d'erreurs détectées durant la phase de chargement. Le processus s'arrête lorsque la limite est atteinte. Par défaut, la valeur limite d'erreurs est de 1000000. Pour plus d'informations, consultez la documentation *Teradata FastLoad Reference*.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Teradata.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Insérer des données dans une table d'une base de données Teradata, page 771*.

tTeradataInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tTeradataInput

Famille de composant	Databases/Teradata	
Fonction	Le composant **tTeradataInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tTeradataInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tSybaseConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.

		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Additional JDBC parameters*	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données, afin de supporter des caractères spécifiques. Exemple : CHARSET=KANJISIS_OS pour supporter les caractères japonais.Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.

	Trim all the String/Char columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Cochez cette case pour supprimer les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Teradata.	

Scénario associé

Pour un scénario associé, consultez les scénarios du composant générique **tDBInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241*

Consultez également le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du **tContextLoad**.

tTeradataMultiLoad

Propriétés du tTeradataMultiLoad

Famille de composant	Databases/Teradata	
Fonction	Le composant **tTeradataMultiLoad** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tTeradataMultiLoad** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (**Main row**).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Execution platform	Sélectionnez le type de système d'exploitation que vous utilisez.
	Database name	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Script generated folder	Parcourez votre répertoire et sélectionnez la destination du fichier qui sera créé.

	Action to data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une clé de recherche sur laquelle baser les opérations Update et Delete. Il est possible de définir les colonnes qui agiront comme clé de recherche à partir du schéma, pour une utilisation de base, ou à partir des options avancées (***Advanced settings***) pour une utilisation optimisée de ces opérations.*
	Where condition in case Delete	Saisissez une condition, qui, si elle est vérifiée, supprimera la ligne.
	Load file	Entrez le nom du fichier à partir duquel vous souhaitez charger les données.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Error file	Parcourez votre répertoire et sélectionnez la destination du fichier dans lesquels seront enregistrés les messages d'erreur.
Advanced settings	*Define Log table*	Cochez cette case pour définir la table de log que vous souhaitez utiliser au lieu de celle par défaut, à savoir celle que vous avez définie dans l'onglet **Basic settings**. La syntaxe requise pour définir la table de log est *databasename.logtablename*.
	BEGIN LOAD	Ce champ vous permet de définir votre commande BEGIN LOAD pour lancer ou relancer une tâche TPump. Vous pouvez spécifiez le nombre de sessions à utiliser, la limite d'erreurs, ainsi que tout autre paramètre nécessaire à l'exécution de la tâche. Pour plus d'informations, consultez la documentation *Teradata MultiLoad Reference*.
	Return mload error	Cochez cette case pour spécifier le code de retour à partir duquel retourner une erreur dans la console.
	Define character set	Spécifiez l'encodage des caractères dont vous avez besoin pour utiliser votre système.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Teradata.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Insérer des données dans une table d'une base de données Teradata, page 771*.

tTeradataOutput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tTeradataOutput

Famille de composant	Databases/Teradata	
Fonction	Le composant **tTeradataOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	**Le tTeradataOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tTeradataConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.

		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table. **Truncate table** : supprimer rapidement le contenu de la table, mais sans possibilité de Rollback.
	Create	Elle n'est pas visible par défaut, sauf si vous choisissez de créer une table à partir de la liste déroulante **Action on table.** La table à créer peut être : **-SET TABLE**: table ne permettant pas de dupliquer les lignes -**MULTI SET TABLE** : table permettant de dupliquer les lignes.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).* La fonction de schéma dynamique peut être utilisée dans différents modes : **Insert**, **Update**, **Insert or Update**, **Update or insert**, **Delete**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.

		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Additional JDBC parameters*	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données, afin de supporter des caractères spécifiques. Exemple : CHARSET=KANJISIS_OS pour supporter les caractères japonais.Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tTeradataOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	Use Batch Size	Nombre de lignes dans chaque lot de traitement de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données Teradata. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.

Scénario associé

Pour un scénario associé, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMysqlOutput**.
- le *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507* du composant **tMysqlOutput**.

tTeradataRollback

Propriétés du tTeradataRollback

Ce composant est étroitement lié aux composants **tTeradataCommit** et **tTeradataConnection**. Ils sont généralement utilisés ensemble lors d'une transaction.

Famille de composant	Databases/Teradata	
Fonction	Le **tTeradataRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant **tTeradataConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec d'autres composants Teradata, notamment les composants **tTeradataConnection** et **tTeradataCommit**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au composant **tTeradataRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522*.

tTeradataRow

Propriétés du tTeradataRow

Famille de composant	Databases/Teradata	
Fonction	Le **tTeradataRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tTeradataRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tTeradataConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.

	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à traiter.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Additional JDBC parameters*	Ajoutez des informations de connexion supplémentaires nécessaires à la connexion à la base de données, afin de supporter des caractères spécifiques. Exemple : CHARSET=KANJISIS_OS pour supporter les caractères japonais.Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**
	Propagate QUERY's recordset	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.

	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des “?” dans l’instruction SQL définie dans le champ **Query** de l’onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l’instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l’écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d’exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénario associé

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l’auto-incrémentation d’une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tTeradataTPump

Propriétés du tTeradataTPump

Famille de composant	Databases/Teradata	
Fonction	Le composant **tTeradataTPump** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tTeradataTPump** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Execution platform	Sélectionnez le type de système d'exploitation que vous utilisez.
	Database name	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Script generated folder	Parcourez votre répertoire et sélectionnez la destination du fichier qui sera créé.

	Action to data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une clé de recherche sur laquelle baser les opérations Update et Delete. Il est possible de définir les colonnes qui agiront comme clé de recherche à partir du schéma, pour une utilisation de base, ou à partir des options avancées (Advanced settings) pour une utilisation optimisée de ces opérations.*
	Where condition in case Delete	Saisissez une condition, qui, si elle est vérifiée, supprimera la ligne.
	Load file	Entrez le nom du fichier à partir duquel vous souhaitez charger les données.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Error file	Parcourez votre répertoire et sélectionnez la destination du fichier dans lesquels seront enregistrés les messages d'erreur.
Advanced settings	*Define Log table*	Cochez cette case pour définir la table de log que vous souhaitez utiliser au lieu de celle par défaut, à savoir celle que vous avez définie dans l'onglet **Basic settings**. La syntaxe requise pour définir la table de log est *databasename.logtablename*.
	BEGIN LOAD	Ce champ vous permet de définir votre commande BEGIN LOAD pour lancer ou relancer une tâche TPump. Vous pouvez spécifiez le nombre de sessions à utiliser, la limite d'erreurs, ainsi que tout autre paramètre nécessaire à l'exécution de la tâche. La valeur par défaut est : `SESSIONS 8 PACK 600 ARRAYSUPPORT ON CHECKPOINT 60 TENACITY 2 ERRLIMIT 1000`. Pour plus d'informations, consultez la documentation *Teradata Parallel Data Pump Reference*.
	Return tpump error	Cochez cette case pour spécifier le code de retour à partir duquel retourner une erreur dans la console.
	Define character set	Spécifiez l'encodage des caractères dont vous avez besoin pour utiliser votre système.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Teradata.	

Scénario : Insérer des données dans une table d'une base de données Teradata

Dans ce scénario, l'objectif est de créer un Job utilisant le composant **tTeradataTPump** pour insérer des données clients dans une table d'une base de données Teradata et retourner la valeur de retour lorsqu'une erreur se produit.

Trois composants sont utilisés pour implémenter cette tâche :

- **tRowGenerator** : génère autant de lignes que nécessaire, en utilisant au hasard des données clients prises dans une liste.
- **tFileOutputDelimited** : écrit les données clients dans un fichier délimité.
- **tTeradataTPump** : insère les données clients dans la table de la base de données Teradata en mode Tpump.

- Déposez les composants suivants de la **Palette** dans l'espace de modélisation graphique : **tRowGenerator**, **tFileOutputDelimited**, **tTeradataTPump**.
- Liez le **tRowGenerator** au **tFileOutputDelimited** à l'aide d'un lien **Row** > **Main**.
- Liez le **tRowGenerator** au **tTeradataTPump** à l'aide d'un lien **Trigger** > **OnSubjobOk**.
- Double-cliquez sur le **tRowGenerator** pour ouvrir la fenêtre **RowGenerator Editor**.
- Dans la fenêtre **tRowGenerator Editor**, définissez les données à générer. Pour ce Job, le schéma se compose de deux colonnes : *ID* et *Name*.

- Dans le champ **Number of Rows for RowGenerator**, saisissez le nombre de lignes à générer, 100 dans cet exemple.
- Double-cliquez sur le **tFileOutputDelimited** afin d'afficher sa vue **Component** et définir ses propriétés.
- A côté du champ **File Name**, parcourez votre répertoire jusqu'au fichier de sortie, ou saisissez un nom pour le fichier à créer.
- Entre guillemets doubles, saisissez les séparateurs à utiliser, à côté des champs **Row Separator** et **Field Separator**, par exemple *\n* et un point-virgule.

- Cliquez sur **Edit schema** et vérifiez que le schéma est retrouvé à partir du schéma d'entrée. Au besoin, cliquez sur **Sync Columns**.
- Double-cliquez sur le composant **tTeradataTPump** afin d'ouvrir sa vue **Component**.
- Dans l'onglet **Basic settings** de la vue **Component**, définissez les paramètres du composant **tTeradataTPump**.

- Saisissez le nom de la base de données dans le champ **Database name**, votre identifiant de connexion à la base de données dans le champ **User name**, et votre mot de passe dans le champ **Password**.
- Spécifiez la table dans laquelle insérer les données clients. Dans ce scénario, la table est *mytable*.
- Dans les champs **Script generated folder**, **Load file** et **Error file**, spécifiez le répertoire respectivement au dossier dans lequel vous stockez les fichiers de scripts générés.
- Dans le champ **Load File**, spécifiez le fichier contenant les données clients à insérer.
- Dans le champ **Error file**, spécifiez le fichier contenant les informations concernant les erreurs.
- Dans le champ **Action on data**, sélectionnez l'option **Insert**.
- Appuyez sur **F6** pour exécuter le Job.
- La console de la vue **Run** s'affiche ainsi :

```
Running job 'tpump'...
Starting job tpump at 17:01 19/07/2010.

[statistics] connecting to socket on port 3740
[statistics] connected
[statistics] disconnected
Job tpump ended at 17:02 19/07/2010. [exit code=0]
```

- Double-cliquez sur le composant **tTeradataTPump** pour retourner à sa vue **Component**.

- Dans l'onglet **Advanced settings**, cochez la case **Return tpump error** et saisissez le code de retour à partir duquel retourner une erreur dans la console. Dans cet exemple, saisissez le chiffre *4* et utilisez les valeurs par défaut pour les autres paramètres.

- Appuyez sur **F6** pour exécuter le Job.
- La console de la vue **Run** s'affiche ainsi :

```
Running job 'tpump'...
Starting job tpump at 17:56 19/07/2010.

[statistics] connecting to socket on port 3549
[statistics] connected
[statistics] disconnected
Exception in component tTeradataTPump_1
java.lang.RuntimeException: TPump returned exit code 12
    at
bug.tpump_0_1.tpump.tTeradataTPump_1Process(tpump.java:307)
    at bug.tpump_0_1.tpump.runJobInTOS(tpump.java:504)
    at bug.tpump_0_1.tpump.main(tpump.java:378)
Job tpump ended at 17:56 19/07/2010. [exit code=0]
```

Une erreur se produit et le `TPump` retourne le code de retour, de 12. Si vous avez besoin d'informations détaillées concernant les informations détaillées de l'erreur, vous pouvez ouvrir le ficheir de log stocké dans le répertoire spécifié dans le champd **Error file** et dans l'onglet **Basic settings** de la vue **Component**.

tVectorWiseCommit

Propriétés du tVectorWiseCommit

Ce composant est étroitement lié aux composants **tVectorWiseConnection** et **tVectorWiseRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/VectorWise	
Fonction	Le composant **tVectorWiseCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tVectorWiseConnection** dans la liste.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tVectorWiseCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants VectorWise et notamment avec les composants **tVectorWiseConnection** et **tVectorWiseRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tVectorWiseConnection** et **tVectorWiseRollback**. Ils sont généralement utilisés avec un composant **tVectorWiseConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tVectorWiseCommit**, consultez *tVerticaConnection, page 794*.

tVectorWiseConnection

Propriétés du tVectorWiseConnection

Ce composant est étroitement lié aux composants **tVectorWiseCommit** et **tVectorWiseRollback**. Ils sont généralemet utilisés avec un composant **tVectorWiseConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/VectorWise	
Fonction	Le composant **tVectorWiseConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Server	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*Auto Commit*	Cochez cette case pour que, lorsqu'une déclaration est terminée, la méthode commit soit appelée automatiquement sur cette déclaration.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants VectorWise, notamment les composants **tVectorWiseCommit** et **tVectorWiseRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tVectorWiseCommit** et **tVectorWiseRollback**. Ils sont généralement utilisés avec un composant **tVectorWiseConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tVectorWiseConnection**, consultez *tMysqlConnection, page 471*

tVectorWiseInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tVectorWiseInput

Famille de composant	Databases/VectorWise	
Fonction	Le composant **tVectorWiseInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tVectorWiseInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (Main row).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case et sélectionnez le composant **tVectorWiseConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Server	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.

		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Table name	Saisissez le nom de la table.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
	Guess Query	Cliquez sur le bouton **Guess Query** pour générer la requête correspondant au schéma de votre table dans le champ **Query**.
	Guess schema	Cliquez sur le bouton pour récupérer le schéma de la table.
Advanced settings	*Trim all the String/Char columns*	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Dans la colonne **Column**, saisissez le nom de la colonne dans laquelle supprimer les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données VectorWise.	

Scénario associé

Pour un scénario associé, consultez les scénarios du composant générique **tDBInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241*

Consultez également le *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190* du **tContextLoad**.

tVectorWiseOutput

Propriétés du tVectorWiseOutput

Famille de composant	Databases/VectorWise	
Fonction	Le composant **tVectorWiseOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	**Le tVectorWiseOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.

	Use an existing connection	Cochez cette case et sélectionnez le composant **tVectorWiseConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.

	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tTeradataOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus de d'écriture dans la base de données.
	Support null in "SQL WHERE" statement	Cochez cette case pour prendre en compte les valeurs Null d'une table de base de données. Assurez-vous que la case Nullable est bien cochée pour les colonnes du schéma correspondantes.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données VectorWise. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénario associé

Pour un scénario associé, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput**.
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMySQLOutput**.

tVectorWiseRollback

Propriétés du tVectorWiseRollback

Ce composant est étroitement lié aux composants **tVectorWiseCommit** et **tVectorWiseConnection**. Ils sont généralement utilisés ensemble lors d'une transaction.

Famille de composant	Databases/VectorWise	
Fonction	Le **tVectorWiseRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant **tVectorWiseConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec d'autres composants VectorWise, notamment les composants **tVectorWiseConnection** et **tVectorWiseCommit**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au composant **tVectorWiseRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522*.

tVectorWiseRow

Propriétés du tVectorWiseRow

Famille de composant	Databases/VectorWise	
Fonction	Le **tVectorWiseRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tVectorWiseRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection	Cochez cette case et sélectionnez le composant **tVectorWiseConnection** adéquat à partir de la liste Component pour réutiliser les paramètres d'une connexion que vous avez déjà définis. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.

	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Table Name	Nom de la table à traiter.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Guess Query	Cliquez sur le bouton **Guess Query** pour générer la requête correspondant au schéma de votre table dans le champ **Query**.
	Query	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.

	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des “?” dans l’instruction SQL définie dans le champ **Query** de l’onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l’instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.
	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l’écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d’exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénario associé

Pour un scénario associé, consultez :

- le *Scénario : Réinitialiser l’auto-incrémentation d’une base de données, page 249* du composant **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du composant **tMySQLRow**.

tVerticaBulkExec

Propriétés du tVerticaBulkExec

Les composants **tVerticaOutputBulk** et **tVerticaBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tVerticaOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le changement des données dans la base de données.

Famille de composant	Databases/Vertica	
Fonction	Le composant **tVerticaBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tVerticaBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données Vertica.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	DB Version	Sélectionnez dans la liste la version de Vertica que vous utilisez.
	Use an existing connection	Cochez cette case lorsque vous utilisez un composant **tVerticaConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données

	Port	Numéro du port d'écoute du serveur de base de données
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois.
	Schema et *Edit schema*	Dans une procédure stockée, le schéma est un paramètre d'entrée. Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Remote Filename	Nom du fichier à traiter. Voir également :*Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Write to ROS (Read Optimized Store)*	Cochez cette case pour stocker les données dans une mémoire physique, et ainsi optimiser la lecture de ces données, puisqu'elles seront compressées et pré-triées.
	Exit job if no row was loaded	Le Job s'arrête automatiquement si aucune ligne n'a été chargée.
	Fields terminated by	Caractère, chaîne ou expression régulière séparant les champs.
	Null string	Chaîne de caractères affichée pour signifier que la valeur est nulle.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant est généralement utilisé avec un composant **tVerticaOutputBulk**. Ensemble, ils offrent un gain de performance important pour l'alimentation d'une base de données Vertica.

Scénarios associés

Pour un scénario associé au composant **tVerticaBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du composant **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du composant **tMysqlOutputBulkExec**.
- le *Scénario : Supprimer et insérer des données dans une base Oracle, page 574* du composant **tOracleBulkExec**.

tVerticaClose

Propriétés du tVerticaClose

Famille de composant	Databases/Vertica	
Fonction	Le composant **tVerticaClose** ferme la connexion à une base de données connectée.	
Objectif	Ce composant permet de fermer une connexion à la base de données Vertica.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tVerticaConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tVerticaConnection** et **tVerticaRollback**. Il est généralement utilisé avec un composant **tVerticaConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tVerticaClose**, consultez *tMysqlConnection, page 471*

tVerticaCommit

Propriétés du tVerticaCommit

Ce composant est étroitement lié aux composants **tVerticaConnection** et **tVerticaRollback**. Ils sont généralement utilisés ensemble lors de transactions.

Famille de composant	Databases/Vertica	
Fonction	Le composant **tVerticaCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tVerticaConnection** dans la liste.
	Close connection	Cette option est cochée par défaut. Elle permet de fermer la connexion à la base de données une fois le commit effectué. Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche. *Si vous utilisez un lien de type* **Row** > **Main** *pour relier le* **tVerticaCommit** *à votre Job, vos données seront commitées ligne par ligne. Dans ce cas, ne cochez pas la case* **Close connection** *car la connexion sera fermée avant la fin du commit de votre première ligne.*
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants Vertica et notamment avec les composants **tVerticaConnection** et **tVerticaRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tVerticaConnection** et **tVerticaRollback**. Ils sont généralement utilisés avec un composant **tVerticaConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tVerticaCommit**, consultez *tVerticaConnection, page 794*.

tVerticaConnection

Propriétés du tVerticaConnection

Ce composant est étroitement lié aux composants **tVerticaCommit** et **tVerticaRollback**. Ils sont généralemet utilisés avec un composant **tVerticaConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Famille de composant	Databases/Vertica	
Fonction	Le composant **tVerticaConnection** ouvre une connexion vers une base de données afin d'effectuer une transaction.	
Objectif	Ce composant permet de commiter les données d'un Job entier en une seule transaction vers une base de données de sortie.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	DB Version	Sélectionnez dans la liste la version de Vertica que vous utilisez.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Table Schema	Nom du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Use or register a shared DB Connection	Cochez cette case pour partager votre connexion ou récupérer une connexion mise en partage dans un Job père ou un Job fils. Cette option vous permet de partager une seule connexion à une base de données dans plusieurs composants de connexion à des bases de données, dans différents niveaux de Jobs, qui peuvent être des Jobs parent ou enfant. **Shared DB Connection Name :** définissez ou saisissez le nom de la connexion partagée.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec des composants Vertica, notamment les composants **tVerticaCommit** et **tVerticaRollback**.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tVerticaCommit** et **tVerticaRollback**. Ils sont généralement utilisés avec un composant **tVerticaConnection** car il permet d'ouvrir une connexion pour la transaction en cours.

Pour un scénario associé au composant **tVerticaConnection**, consultez *tMysqlConnection, page 471*

tVerticaInput

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tVerticaInput

Famille de composant	Databases/Vertica	
Fonction	Le composant **tVerticaInput** lit une base de données et en extrait des champs à l'aide de requêtes.	
Objectif	Le **tVerticaInput** exécute une requête en base de données selon un ordre strict qui doit correspondre à celui défini dans le schéma. La liste des champs récupérée est ensuite transmise au composant suivant via une connexion de flux (**Main row**).	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio** .
	DB Version	Sélectionnez dans la liste la version de Vertica que vous utilisez.

	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tVerticaConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant (**Repository**). Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.

	Table Name	Nom de la table à lire.
	Query type et *Query*	Saisissez votre requête de base de données en faisant attention à ce que l'ordre des champs corresponde à celui défini dans le schéma. *Si vous utilisez la fonction de schéma dynamique, la requête SELECT doit inclure le caractère *, afin de récupérer toutes les colonnes de la table sélectionnée.*
Advanced settings	*Trim all the String/Char columns*	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes contenant des chaînes de caractères.
	Trim column	Supprimez les espaces en début et en fin de champ dans les colonnes sélectionnées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant couvre toutes les possibilités de requête SQL dans les bases de données Vertica.	

Scénarios associés

Voir également les scénarios du **tDBInput** :

- *Scénario 1 : Afficher les données sélectionnées à partir d'une table, page 239.*
- *Scénario 2 : Utiliser la variable StoreSQLQuery, page 241.*

Voir également le scénario du **tContextLoad,** *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190.*

tVerticaOutput

Propriétés du tVerticaOutput

Famille de composant	Databases/Vertica	
Fonction	Le composant **tVerticaOutput** écrit, met à jour, modifie ou supprime les données d'une base de données.	
Objectif	Le **tVerticaOutput** exécute l'action définie sur la table et/ou sur les données d'une table, en fonction du flux entrant provenant du composant précédent.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de configuration de connexion à la base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	DB Version	Sélectionnez dans la liste la version de Vertica que vous utilisez.

	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tVerticaConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données
	Table	Nom de la table à créer. Vous ne pouvez créer qu'une seule table à la fois.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Drop a table if exists and create** : supprimer la table si elle existe déjà, puis en créer une nouvelle. **Clear a table** : supprimer le contenu de la table.

	Action on data	Vous pouvez effectuer les opérations suivantes sur les données de la table sélectionnée : **Insert** : Ajouter de nouvelles entrées à la table. Le Job s'arrête lorsqu'il détecte des doublons. **Update** : Mettre à jour les entrées existantes. **Insert or update** : Ajouter des entrées ou mettre à jour les entrées existantes. **Update or insert** : Mettre à jour les entrées existantes ou en créer si elles n'existent pas. **Delete** : Supprimer les entrées correspondantes au flux d'entrée. *Il est nécessaire de spécifier au minimum une colonne comme clé primaire sur laquelle baser les opérations* **Update** *et* **Delete**. *Pour cela, cliquez sur le bouton* **[...]** *à côté du champ* **Edit Schema** *et cochez la ou les case(s) correspondant à la ou aux colonne(s) que vous souhaitez définir comme clé(s) primaire(s). Pour une utilisation avancée, cliquez sur l'onglet* **Advanced settings** *pour définir simultanément les clés primaires sur lesquelles baser les opérations de mise à jour (Update) et de suppression (Delete). Pour cela, cochez la case* **Use field options** *et sélectionnez la case* **Key in update** *correspondant à la colonne sur laquelle baser votre opération de mise à jour (Update). Procédez de la même manière avec les cases* **Key in delete** *pour les opérations de suppression (Delete).*
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Commit every*	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.

	Use Batch Mode	Cochez cette case afin de paramétrer le nombre de lignes dans chaque lot de traitement de données, dans le champ **Batch Size**.
	Additional Columns	Cette option n'est pas disponible si vous venez de créer la table de données (que vous l'ayez préalablement supprimée ou non). Cette option vous permet d'effectuer des actions sur les colonnes, à l'exclusion des actions d'insertion, de mise à jour, de suppression ou qui nécessitent un prétraitement particulier.
		Name : Saisissez le nom de la colonne à modifier ou à insérer.
		SQL expression : Saisissez la déclaration SQL à exécuter pour modifier ou insérer les données dans les colonnes correspondantes.
		Position : Sélectionnez **Before**, **Replace** ou **After**, en fonction de l'action à effectuer sur la colonne de référence.
		Reference column : Saisissez une colonne de référence que le composant **tVerticaOutput** peut utiliser pour situer ou remplacer la nouvelle colonne ou celle à modifier.
	Use field options	Cochez cette case pour personnaliser une requête, surtout lorsqu'il y a plusieurs actions sur les données.
	Enable debug mode	Cochez cette case pour afficher chaque étape du processus d'écriture dans la base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités des requêtes SQL. Il permet de faire des actions sur une table ou les données d'une table d'une base de données Vertica. Il permet aussi de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données en erreur. Pour un exemple d'utilisation, consultez le *Scénario 3 : Récupérer les données erronées à l'aide d'un lien Reject, page 501* du composant **tMysqlOutput**.	

Scénarios associés

Pour un scénario associé au composant **tVerticaOutput**, consultez :

- le *Scénario : Afficher la base de données en sortie, page 245* du composant **tDBOutput.**
- le *Scénario 1 : Ajouter une colonne et modifier les données, page 493* du composant **tMySQLOutput**.

tVerticaOutputBulk

Propriétés du tVerticaOutputBulk

Les composants **tVerticaOutputBulk** et **tVerticaBulkExec** sont généralement utilisés ensemble pour d'une part générer en sortie le fichier qui sera d'autre part utilisé comme paramètre dans l'exécution de la requête SQL énoncée. Cette exécution en deux étapes est unifiée dans le composant **tVerticaOutputBulkExec**, détaillé dans une section séparée. L'intérêt de proposer deux composants séparés réside dans le fait que cela permet de procéder à des transformations avant le chargement des données dans la base de données.

Famille de composant	Databases/Vertica	
Fonction	Ecrit un fichier composé de colonnes et basé sur le délimiteur défini et sur les standards Vertica.	
Objectif	Prépare le fichier à utiliser comme paramètre dans la requête INSERT servant à alimenter une base de données Vertica.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
Advanced settings	*Row separator*	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Include header	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.

	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec le composant **tVerticaBulkExec**. Ensemble, ils offrent un gain de performance important pour l'alimentation d'une base de données Vertica.	

Scénarios associés

Pour un scénario associé au **tVerticaOutputBulk**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du composant **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.

tVerticaOutputBulkExec

Propriétés du tVerticaOutputBulkExec

Les composants **tVerticaOutputBulk** et **tVerticaBulkExec** sont généralement utilisés ensemble comme deux parties d'un processus en deux étapes. Dans la première étape, un fichier de sortie est généré. Dans la deuxième étape, ce fichier est utilisé lors de l'opération d'INSERT afin de populer une base de données. Cette exécution en deux étapes est unifiée dans le composant **tVerticaOutputBulkExec**.

Famille de composant	Databases/Vertica	
Fonction	Le composant **tVerticaOutputBulkExec** effectue une action d'Insert sur les données fournies.	
Objectif	Le **tVerticaOutputBulkExec** est un composant dédié qui permet un gain de performance pendant les opérations d'Insert dans une base de données Vertica.	
Basic settings	*Property Type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	DB Version	Sélectionnez dans la liste la version de Vertica que vous utilisez.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	DB Name	Nom de la base de données.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Table	Nom de la table à écrire. Notez qu'une seule table peut être écrite à la fois et la table doit déjà exister pour que l'opération d'insert soit autorisée.
	Action on table	Vous pouvez effectuer l'une des opérations suivantes sur les données de la table sélectionnée : **None** : n'effectuer aucune opération de table. **Drop and create the table** : supprimer la table puis en créer une nouvelle. **Create a table** : créer une table qui n'existe pas encore. **Create table if doesn't exist** : créer la table si nécessaire. **Clear a table** : supprimer le contenu de la table.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.

		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	File Name	Nom du fichier à traiter. Voir également :*Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
Advanced settings	*Write to ROS (Read Optimized Store)*	Cochez cette case pour stocker les données dans une mémoire physique, et ainsi optimiser la lecture de ces données, puisqu'elles seront compressées et pré-triées.
	Exit job if no row was loaded	Le Job s'arrête automatiquement si aucune ligne n'a été chargée.
	Field Separator	Caractère, chaîne ou expression régulière séparant les champs.
	Null string	Chaîne de caractères affichée pour signifier que la valeur est nulle.
	Include header	Cochez cette case pour inclure l'en-tête des colonnes dans le fichier.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est principalement utilisé lorsqu'aucune transformation particulière n'est requise sur les données à charger dans la base de données.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé au **tVerticaOutputBulkExec**, consultez :

- le *Scénario : Insérer des données transformées dans une base MySQL, page 514* du composant **tMysqlOutputBulk**.
- le *Scénario : Insérer des données dans une base MySQL, page 520* du **tMysqlOutputBulkExec**.

tVerticaRollback

Propriétés du tVerticaRollback

Ce composant est étroitement lié aux composants **tVerticaCommit** et **tVerticaConnection**. Ils sont généralement utilisés ensemble lors d'une transaction.

Famille de composant	Databases/Vertica	
Fonction	Le **tVerticaRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Component list*	Sélectionnez le composant **tVerticaConnection** dans la liste s'il y a plus d'une connexion dans votre Job.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé avec d'autres composants Oracle, notamment les composants **tVerticaConnection** et **tVerticaCommit**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé au composant **tVerticaRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522*.

tVerticaRow

Propriétés du tVerticaRow

Famille de composant	Databases/Vertica	
Fonction	Le **tVerticaRow** est le composant spécifique à ce type de base de données. Il exécute des requêtes SQL déclarées sur la base de données spécifiée. Le suffixe Row signifie que le composant met en place un flux dans le Job bien que ce composant ne produise pas de données en sortie.	
Objectif	Selon la nature de la requête et de la base de données, **tVerticaRow** agit sur la structure même de la base de données ou sur les données (mais sans les manipuler). Le SQLBuilder peut vous aider à rapidement et aisément écrire vos requêtes.	
Basic settings	*Property type*	Peut être Built-in ou **Repository**.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	DB version	Sélectionnez la version de Vertica que vous utilisez.
	Use an existing connection	Cochez cette case lorsque vous utilisez le composant **tVerticaConnection**. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afin de paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur de base de données.
	Port	Numéro du port d'écoute du serveur de base de données.
	Database	Nom de la base de données.

	Schema	Nom exact du schéma.
	Username et *Password*	Informations d'authentification de l'utilisateur de base de données.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Table name	Nom de la table à lire.
	Query type	Peut être Built-in ou Repository.
		Built-in : Saisissez manuellement votre requête ou construisez-la à l'aide de SQLBuilder.
		Repository : Sélectionnez la requête appropriée dans le **Repository**. Le champ **Query** est renseigné automatiquement.
	Query	Saisissez votre requête en faisant particulièrement attention à l'ordre des champs afin qu'ils correspondent à la définition du schéma.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Propagate QUERY's recordset*	Cochez cette case pour insérer les résultats de la requête dans une colonne du flux en cours. Sélectionnez cette colonne dans la liste **use column**.
	Use PreparedStatement	Cochez cette case pour utiliser une instance PreparedStatement afin de requêter votre base de données. Dans le tableau **Set PreparedStatement Parameter**, définissez les valeurs des paramètres représentés par des "?" dans l'instruction SQL définie dans le champ **Query** de l'onglet **Basic settings**. **Parameter Index** : Saisissez la position du paramètre dans l'instruction SQL. **Parameter Type** : Saisissez le type du paramètre. **Parameter Value** : Saisissez la valeur du paramètre. Cette option est très utile si vous devez effectuer de nombreuses fois la même requête. Elle permet un gain de performance.

	Commit every	Nombre de lignes à inclure dans le lot avant de commencer l'écriture dans la base. Cette option garantit la qualité de la transaction (cependant pas de rollback) et surtout une meilleure performance d'exécution.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant offre la flexibilité des requêtes sur les bases de données et couvre toutes les possibilités de requêtes SQL.	

Scénario associé

Pour un scénario associé au composant **tVerticaRow**, consultez :

- le *Scénario : Réinitialiser l'auto-incrémentation d'une base de données, page 249* du **tDBSQLRow**.
- le *Scénario 1 : Supprimer et re-générer un index de table MySQL, page 526* du **tMySQLRow**.

Composants ELT

Ce chapitre passe en revue les propriétés des principaux composants présents dans la famille **ELT** de la **Palette** de **Talend Open Studio**.

La famille ELT regroupe les connecteurs de bases de données les plus courants, ainsi que des composants de traitement, tous dédiés au mode ELT dans lequel les SGBD cibles deviennent des outils de transformation.

Ce mode supporte les bases de données les plus courantes, comme Teradata, Oracle, Vertica, Netezza, Sybase, etc.

tELTAggregate

Propriétés du tELTAggregate

Famille de composant	ELT	
Fonction	Le composant **tELTAggregate** collecte les données d'une ou plusieurs colonnes et gère ces données en un seul bloc. Ce composant exécute en temps réel des transformations de données dans la SGBD elle-même.	
Objectif	Il permet d'établir des métriques basées sur des valeurs ou des calculs.	
Basic settings	*Component list*	Sélectionnez le composant de connexion à la base de données dans la liste si vous utilisez plus d'une connexion dans le Job en cours.
	Database name	Nom de la base de données.
	Source table name	Nom de la table contenant les données dont vous souhaitez collecter les données.
	Target table name	Nom de la table dans laquelle vous souhaitez écrire les données collectées et transformées.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Operations	Sélectionnez le type d'opération et la valeur à utiliser pour le calcul et le champ de sortie.
		Output Column : Sélectionnez le champ de destination dans la liste.
		Function : Sélectionnez une des opérations suivantes à effectuer sur les données : **count :** calcule le nombre de lignes, **min :** sélectionne la plus petite valeur, **max :** sélectionne la plus grande valeur, **avg :** calcule la moyenne, **sum :** calcule la somme, **count (distinct) :** compte le nombre de lignes sans les doublons.

		Input column position : Sélectionnez la colonne d'entrée à partir de laquelle vous souhaitez collecter les données à agréger.
	Group by	Définit les ensembles d'agrégation, dont les valeurs sont utilisées pour les calculs.
		Output Column : Sélectionnez le libellé de colonne dans la liste fournie, basée sur la structure de schéma que vous avez définie. Vous pouvez ajouter autant de colonnes de sortie que vous le souhaitez afin d'affiner les agrégations.
		Input Column position : Faites la correspondance entre les libellés des colonnes d'entrée avec ceux des colonnes de sortie, dans le cas où vous souhaitez que les libellés du schéma de sortie soient différents du schéma d'entrée.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
SQL Template	*SQLTemplate List*	Cliquez sur le bouton **Add** pour ajouter un ou plusieurs modèle SQL du système à la liste **SQLTemplateList**. Cliquez dans la cellule de la colonne **SQLTemplate List** que vous venez d'ajouter et sélectionnez le modèle SQL du système adéquat dans la liste de modèles fournis par **Talend**. **Note :** Vous pouvez créer vos propres modèles SQL et les ajouter à la liste **SQLTemplate List**. Pour créer un nouveau modèle SQL : 1- Sélectionnez un modèle déjà existant dans la liste **SQLTemplate list** et cliquez dans son code dans le champ de code. Une boîte de dialogue s'ouvrira vous demandant de créer un nouveau modèle. 2- Cliquez sur **Yes** pour ouvrir la boîte de dialogue **New SQLTemplate**. 3- Renseignez les informations de votre modèle SQL et cliquez sur **Finish** pour fermer la boîte de dialogue. Un éditeur de modèle SQL apparaît dans lequel vous pouvez saisir votre code. 4- Cliquez sur le bouton **Add** pour ajouter le nouveau modèle dans la liste **SQLTemplate list**. Pour plus d'informations, consultez la section *Utilisation des SQL Templates, page 159* du **Guide Utilisateur** **Talend Open Studio**.
Utilisation	Ce composant est un composant intermédiaire. Vous pouvez l'utiliser avec d'autres composants de base de données, notamment les composants de connexion et de commit.	
Limitation	n/a	

Scénario : Filtrer et agréger les colonnes d'une table directement dans le SGBD

Le scénario Java suivant crée un Job qui ouvre une connexion à une base de données Mysql et :

- crée un schéma à partir de la table d'une base de données, les lignes de ce schéma correspondent aux noms des colonnes spécifiées dans le filtre,
- filtre les colonnes de cette même table afin de n'obtenir que les données correspondant aux colonnes filtrées,
- collecte les données de la colonne filtrée, regroupées en fonction d'une valeur spécifique et écrit les données agrégées dans la table de cible.

Pour filtrer et agréger les colonnes de la table de votre base de données :

- Cliquez et déposez les composants suivants dans l'espace de modélisation : **tELTMysqlconnection**, **tELTFilterColumns**, **tELTFilterRows**, **tELTAggregate**, **tELTCommit** et **tELTRollback**.
- Connectez les cinq composants à l'aide de liens de type **OnComponentOk**.
- Connectez les composants **tELTAggregate** et **tELTRollback** à l'aide d'un lien de type **OnComponentError**.

- Dans l'espace de modélisation, sélectionnez le **tMysqlConnection** et cliquez sur la vue **Component** pour paramétrer ses propriétés.
- Dans l'onglet **Basic settings**, renseignez manuellement les informations de connexion ou sélectionnez-les à partir des variables de contexte, via la raccourci **Ctrl+Espace** cliquez sur le champ correspondant si les vous avez stocké sous le noeud **DB connection** du répertoire **Metadata** du **Repository**.

Pour plus d'informations sur les métadonnées, consultez *Centraliser les métadonnées* du **Guide utilisateur** de **Talend Open Studio**.

- Dans l'espace de modélisation, sélectionnez le composant **tELTFilterColumns** et cliquez sur la vue **Component** pour paramétrer ses propriétés.

- Dans la liste déroulante **Database type**, sélectionnez la base de données adéquate.
- Dans la liste déroulante **Component list**, sélectionnez le composant de connexion à la base de données si plusieurs composants de connexion sont utilisés dans votre Job.
- Dans le champ **Database name**, saisissez le nom de la base de données à laquelle vous vous êtes connecté.
- Dans le champ **Source table name**, saisissez le nom de votre table source contenant les données que vous souhaitez traiter et cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour définir la structure des données de votre table source.
- Dans le champ **Target table name**, saisissez le nom de la table cible contenant vos nouvelles données traitées et cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour définir la structure de la table cible.

Lorsque vous définissez la structure des données de la table source, le nom des colonnes apparaît automatiquement dans la colonne **Column** du tableau **Column filters**.

Dans ce scénario, la table source contient les cinq colonnes : *id*, *First_Name*, *Last_Name*, *Address* et *id_State*.

- Dans le tableau **Column filters**, filtrez les colonnes en cochant la case des colonnes que vous souhaitez écrire dans votre table cible.

Dans ce scénario, le composant **tELTFilterColumns** ne garde que les colonnes *id*, *First_Name* et *id_State* de la table source.

Dans la vue **Component**, vous pouvez cliquer sur l'onglet **SQL Template** et ajouter des modèles SQL du système ou créer vos propres modèles SQL et les utilisez dans votre Job afin d'effectuer les opérations que vous souhaitez. Pour plus d'informations, consultez *Propriétés du tELTFilterColumns, page 821*.

- Dans l'espace de modélisation, sélectionnez le composant **tELTFilterRows** et cliquez sur l'onglet **Component** pour paramétrer ses propriétés.

- Dans la liste déroulante **Database type**, sélectionnez la base de données adéquate.
- Dans la liste déroulante **Component list**, sélectionnez le composant de connexion à la base de données si plusieurs composants de connexion sont utilisés dans votre Job.
- Dans le champ **Database name**, saisissez le nom de la base de données à laquelle vous vous êtes connecté.
- Dans le champ **Source table name**, saisissez le nom de votre table source contenant les données que vous souhaitez traiter et cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour définir la structure des données de votre table source.
- Dans le champ **Target table name**, saisissez le nom de la table cible contenant vos nouvelles données traitées et cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour définir la structure de la table cible.

Dans ce scénario, la table source et la table cible contiennent les trois colonnes *filtrées id*, *First_Name* et *id_State*.

- Dans le champ **Where condition**, saisissez la commande WHERE vous permettant de n'extraire que les champs correspondant à vos critères.

Dans ce scénario, le composant **tELTFilterRows** filtre la colonne *First_Name* de la table source pour n'obtenir que les prénoms commencent par la lettre "a".

- Dans l'espace de modélisation, sélectionnez le composant **tELTAggregate** et cliquez sur la vue **Component** pour paramétrer ses propriétés.
- Dans la liste déroulante **Database type**, sélectionnez la base de données adéquate.
- Dans la liste **Component list**, sélectionnez le composant de connexion à la base de données si plusieurs composants de connexion sont utilisés dans votre Job.

- Dans le champ **Database name**, saisissez le nom de la base de données à laquelle vous vous êtes connecté.
- Dans le champ **Source table name**, saisissez le nom de votre table source contenant les données que vous souhaitez traiter et cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour définir la structure des données de votre table source.
- Dans le champ **Target table name**, saisissez le nom de la table cible contenant vos nouvelles données traitées et cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour définir la structure de la table cible.

Le schéma de la table source est composé de trois colonnes : *id*, *First_Name* et *id_State*. Le schéma de la table cible est composé de deux colonnes : *customers_status* et *customers_number*. Dans ce scénario, vous souhaitez regrouper les clients par statut marital et compter le nombre de client pour chaque groupe marital. Pour cela, définissez les tableaux **Operations** et **Grouped by**.

- Dans le tableau **Operations**, cliquez sur le bouton **[+]** pour ajouter une ou plusieurs lignes et cliquez dans la cellule de la colonne **Output column** et sélectionnez la colonne de sortie qui contiendra les données à compter.
- Cliquez dans la cellule de la colonne **Function** et sélectionnez l'opération à effectuer.

- Dans le tableau **Group by**, cliquez sur le bouton **[+]** pour ajouter une ou plusieurs lignes et cliquez dans la cellule de la colonne **Output column** pour sélectionner la colonne de sortie qui contiendra les données agrégées.
- Dans l'espace de modélisation, sélectionnez le composant **tELTCommit** et cliquez sur la vue **Component** pour paramétrer ses propriétés.
- Dans la liste déroulante **Database type**, sélectionnez la base de données adéquate.
- Dans la liste déroulante **Component list**, sélectionnez le composant de connexion à la base de données si plusieurs composants de connexion sont utilisés dans votre Job.
- Effectuez la même procédure pour le composant **tELTRollback**.
- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

La table *aggregate_customers* composée de deux colonnes est créée dans la base de données. Elle regroupe les clients en fonction de leur statut marital et compte le nombre de client pour chaque statut marital.

tELTCommit

Propriétés du tELTCommit

Ce composant est étroitement lié au composant **tELTRollback** et au composant ELT de connexion à la base de données sur laquelle vous travaillez. Ils sont généralement utilisés ensemble lors de transactions.

Famille du composant	Databases/ELT	
Fonction	Le composant **tELTCommit** valide les données traitées dans un Job à partir d'une base de données connectée.	
Objectif	En utilisant une connexion unique, commitez en une seule fois une transaction globale au lieu de commiter chaque ligne ou chaque lot de lignes. Ce composant permet un gain de performance.	
Basic settings	*Database Type*	Sélectionnez dans le menu déroulant le type de base de données sur lequel vous voulez travailler.
	Component list	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant de connexion à la base de données que vous utilisez dans la liste.
	Close connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
SQL Template	*SQLTemplate List*	Cliquez sur le bouton **Add** pour ajouter un ou plusieurs modèle SQL du système à la liste **SQLTemplateList**. Cliquez dans la cellule de la colonne **SQLTemplate List** que vous venez d'ajouter et sélectionnez le modèle SQL du système adéquat dans la liste de modèles fournis par **Talend**. **Note :** Vous pouvez créer vos propres modèles SQL et les ajouter à la liste **SQLTemplate List**. Pour créer un nouveau modèle SQL : 1- Sélectionnez un modèle déjà existant dans la liste **SQLTemplate list** et cliquez dans son code dans le champ de code. Une boîte de dialogue s'ouvrira vous demandant de créer un nouveau modèle. 2- Cliquez sur **Yes** pour ouvrir la boîte de dialogue **New SQLTemplate**. 3- Renseignez les informations de votre modèle SQL et cliquez sur **Finish** pour fermer la boîte de dialogue. Un éditeur de modèle SQL apparaît dans lequel vous pouvez saisir votre code. 4- Cliquez sur le bouton **Add** pour ajouter le nouveau modèle dans la liste **SQLTemplate list**. Pour plus d'informations, consultez la section *Utilisation des SQL Templates, page 159* du **Guide Utilisateur** Talend Open Studio.

Utilisation	Ce composant est généralement utilisé avec des composants ELT et notamment avec le composant **tELTRollback,** ainsi qu'avec le composant ELT de connexion à la base de données que vous utilisez.
Limitation	n/a

Scénario associé

Ce composant est étroitement lié au composant **tELTRollback** et au composant ELT de connexion à la base de données avec laquelle vous travaillez. Ils sont généralement utilisés avec le composant ELT de connexion correspondant car il permet d'ouvrir une connexion pour la transaction en cours.

Pour plus d'informations relatives au fonctionnement du composant **tELTCommit**, consultez *tVerticaConnection, page 794.*

tELTFilterColumns

Propriétés du tELTFilterColumns

Famille de composant	ELT	
Fonction	Le composant **tELTFilterColumns** modifie le schéma de la table d'une base de données en filtrant les colonnes. Ce composant filtre en temps réel les données dans la SGBD elle-même.	
Objectif	Il permet d'homogénéiser des schémas en réorganisant ou en supprimant des colonnes ou en ajoutant de nouvelles colonnes.	
Basic settings	*Database Type*	Sélectionnez dans le menu déroulant le type de base de données sur lequel vous voulez travailler.
	Component list	Sélectionnez le composant de connexion à la base de données si vous utilisez plus d'une connexion dans votre Job en cours.
	Database name	Nom de la base de données.
	Source table name	Nom de la table contenant les données à filtrer.
	Target table name	Nom de la table dans laquelle vous souhaitez écrire les données filtrées.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Column filters	Dans le tableau, cochez la case **Filter** pour filtrer toutes vos colonnes, ou, pour en filtrer moins, cliquez sur la ou les case(s) en face du nom de la ou des colonne(s).
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.

SQL Template	*SQLTemplate List*	Cliquez sur le bouton **Add** pour ajouter un ou plusieurs modèle SQL du système à la liste **SQLTemplateList**. Cliquez dans la cellule de la colonne **SQLTemplate List** que vous venez d'ajouter et sélectionnez le modèle SQL du système adéquat dans la liste de modèles fournis par **Talend**. **Note :** Vous pouvez créer vos propres modèles SQL et les ajouter à la liste **SQLTemplate List**. Pour créer un nouveau modèle SQL : 1- Sélectionnez un modèle déjà existant dans la liste **SQLTemplate list** et cliquez dans son code dans le champ de code. Une boîte de dialogue s'ouvrira vous demandant de créer un nouveau modèle. 2- Cliquez sur **Yes** pour ouvrir la boîte de dialogue **New SQLTemplate**. 3- Renseignez les informations de votre modèle SQL et cliquez sur **Finish** pour fermer la boîte de dialogue. Un éditeur de modèle SQL apparaît dans lequel vous pouvez saisir votre code. 4- Cliquez sur le bouton **Add** pour ajouter le nouveau modèle dans la liste **SQLTemplate list**. Pour plus d'informations, consultez la section *Utilisation des SQL Templates, page 159* du **Guide Utilisateur** **Talend Open Studio**.
Utilisation	Ce composant est un composant intermédiaire. Vous pouvez l'utiliser avec d'autres composants de base de données, notamment les composants de connexion et de commit.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Filtrer et agréger les colonnes d'une table directement dans le SGBD, page 814*.

tELTFilterRows

Propriétés du tELTFilterRows

Famille de composant	ELT	
Fonction	Le composant **tELTFilterRows** vous permet de filtrer les lignes d'une table. Ce composant filtre en temps réel les données dans la SGBD elle-même.	
Objectif	Il permet de mettre en place un filtrage de lignes sur n'importe quelle donnée source grâce à une commande WHERE.	
Basic settings	*Database Type*	Sélectionnez dans le menu déroulant le type de base de données sur lequel vous voulez travailler.
	Component List	Sélectionne le composant de connexion à la base de données si vous utilisez plus d'une connexion dans le Job en cours.
	Database name	Nom de la base de données.
	Source table name	Nom de la table contenant les données à filtrer.
	Target table name	Nom de la table dans laquelle vous souhaitez écrire les données filtrées.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Where condition	Utilisez une commande WHERE pour définir un critère à partir duquel vous souhaitez filtrer vos données. Vous pouvez utiliser la commande WHERE pour sélectionner des lignes spécifiques de votre table en fonction de certains critères ou certaines conditions.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.

SQL Template	*SQLTemplate List*	Cliquez sur le bouton **Add** pour ajouter un ou plusieurs modèle SQL du système à la liste **SQLTemplateList**. Cliquez dans la cellule de la colonne **SQLTemplate List** que vous venez d'ajouter et sélectionnez le modèle SQL du système adéquat dans la liste de modèles fournis par **Talend**. **Note :** Vous pouvez créer vos propres modèles SQL et les ajouter à la liste **SQLTemplate List**. Pour créer un nouveau modèle SQL : 1- Sélectionnez un modèle déjà existant dans la liste **SQLTemplate list** et cliquez dans son code dans le champ de code. Une boîte de dialogue s'ouvrira vous demandant de créer un nouveau modèle. 2- Cliquez sur **Yes** pour ouvrir la boîte de dialogue **New SQLTemplate**. 3- Renseignez les informations de votre modèle SQL et cliquez sur **Finish** pour fermer la boîte de dialogue. Un éditeur de modèle SQL apparaît dans lequel vous pouvez saisir votre code. 4- Cliquez sur le bouton **Add** pour ajouter le nouveau modèle dans la liste **SQLTemplate list**. Pour plus d'informations, consultez la section *Utilisation des SQL Templates, page 159* du **Guide Utilisateur Talend Open Studio**.
Utilisation	Ce composant est un composant intermédiaire. Vous pouvez l'utiliser avec d'autres composants de base de données, notamment les composants de connexion et de commit.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Filtrer et agréger les colonnes d'une table directement dans le SGBD, page 814*.

tELTJDBCInput

Propriétés de tELTJDBCInput

Les composants **tELTJDBCInput**, **tELTJDBCOutput**, et **tELTJDBCMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base JDBC afin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille de composant	ELT	
Fonction	Fournit le schéma de la table d'entrée à la commande SQL qui sera exécutée.	
Objectif	Permet d'ajouter autant de tables que nécessaires dans une commande Insert qui peut être complexe.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités. Le schéma est soit local (built-in) soit distant dans le Repository. Le schéma défini est ensuite passé au composant ELT Mapper afin de l'utiliser dans la commande SQL Insert Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Default Table Name	Saisissez le nom de la table par défaut.
	Default Schema Name	Saisissez le nom du schéma par défaut.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	L'utilisation du composant **tELTJDBCInput** est étroitement liée à celle du **tELTJDBCMap**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénarios associés

Pour des scénarios utilisant un composant semblable au **tELTJDBCInput**, consultez les scénarios suivants du composant **tELTMysqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTJDBCMap

Propriétés de tELTJDBCMap

Les composants **tELTJDBClInput**, **tELTJDBCOutput**, et **tELTJDBCMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base JDBC afin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille de composant	ELT	
Fonction	Permet de construire votre commande SQL graphiquement en utilisant la table fournie en entrée.	
Objectif	Utilise les tables fournies en entrée pour alimenter les paramètres nécessaires à l'exécution de la commande SQL souhaitée. La commande peut contenir des jointures internes comme externes entre différentes tables ou entre une table et ses alias.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	ELT Oracle Map Editor	L'éditeur ELT du Mapper vous permet de définir le schéma de sortie ainsi que de construire graphiquement la commande SQL à exécuter.

	Style link	Sélectionnez le type d'affichage des liens. **Auto** : par défaut, les liens entre les schémas d'entrée et de sortie et les paramètres du service Web sont en forme de courbe. **Curves** : les liens entre les schémas et les paramètres du service Web sont en forme de ligne. **Line** : les liens entre les schémas et les paramètres du service Web sont en forme de lignes droites. Cette option vous permettra d'optimiser les performances.
	Property type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données
	Port	Numéro de port d'écoute du serveur de la base.
	Database	Nom de la base de données
	Username et *Password*	Données d'authentification de l'utilisateur de la base
	Encoding Type	Sélectionnez l'encodage dans la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour le traitement des données d'une base de données.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	L'utilisation du composant **tELTJDBCMap** est étroitement liée à celle des composants **tELTJDBCInput** et **tELTJDBCOutput**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénarios associés

Pour des scénarios associés, consultez les scénarios suivants du composant **tELTMysqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTJDBCOutput

Propriétés de tELTJDBCOutput

Les composants **tELTJDBCInput**, **tELTJDBCOutput**, et **tELTJDBCMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base MSSql afin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille du composant	ELT	
Fonction	Effectue l'action sur la table telle que spécifiée et insère les données selon le schéma de sortie défini dans le Mapper ELT.	
Objectif	Exécute la requête Insert dans la base de données JDBC.	
Basic settings	*Action on data*	Sur les données de la table définie, vous pouvez effectuer les opérations suivantes : **Insert** : Ajoute de nouvelles lignes à la table. Si des doublons sont identifiés, le Job s'arrête. Note : dans les composants ELT JDBC, seule l'opération Insert est disponible.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Default Table Name	Saisissez le nom de la table par défaut, entre guillemets doubles.
	Default Schema Name	Saisissez le nom du schéma par défaut, entre guillemets doubles.
	Use different table name	Cochez cette case afin de définir le nom d'une table de sortie différente, entre guillemets doubles, dans le champ **Table name** qui apparaît.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	L'utilisation du composant **tELTJDBCOutput** est étroitement liée à celle des composants **tELTJDBCInput** et **tELTJDBCMap**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.

Scénarios associés

Pour des scénarios utilisant un composant semblable au **tELTJDBCOutput**, consultez les scénarios suivants du composant **tELTJDBCMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTMSSqlInput

Propriétés de tELTMSSqlInput

Les composants **tELTMSSqlInput**, **tELTMSSqlOutput**, et **tELTMSSqlMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base MSSql afin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille de composant	ELT	
Fonction	Fournit le schéma de la table d'entrée à la commande SQL qui sera exécutée.	
Objectif	Permet d'ajouter autant de tables que nécessaires dans une commande Insert qui peut être complexe.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités. Le schéma est soit local (built-in) soit distant dans le Repository. Le schéma défini est ensuite passé au composant ELT Mapper afin de l'utiliser dans la commande SQL Insert Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Default Table Name	Saisissez le nom de la table par défaut.
	Default Schema Name	Saisissez le nom du schéma par défaut.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	L'utilisation du composant **tELTMSSqlInput** est étroitement liée à celle du **tELTMSSqlMap**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénarios associés

Pour des scénarios utilisant un composant semblable au **tELTMSSqlInput**, consultez les scénarios suivants du composant **tELTMysqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTMSSqlMap

Propriétés de tELTMSSqlMap

Les composants **tELTMSSqlInput**, **tELTMSSqlOutput**, et **tELTMSSqlMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base MSSql afin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille de composant	ELT	
Fonction	Permet de construire votre commande SQL graphiquement en utilisant la table fournie en entrée.	
Objectif	Utilise les tables fournies en entrée pour alimenter les paramètres nécessaires à l'exécution de la commande SQL souhaitée. La commande peut contenir des jointures internes comme externes entre différentes tables ou entre une table et ses alias.	
Basic settings	*Use an existing connection*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	ELT MSSql Map Editor	L'éditeur ELT du Mapper vous permet de définir le schéma de sortie ainsi que de construire graphiquement la commande SQL à exécuter.

	Style link	Sélectionnez le type d'affichage des liens. **Auto** : par défaut, les liens entre les schémas d'entrée et de sortie et les paramètres du service Web sont en forme de courbe. **Curves** : les liens entre les schémas et les paramètres du service Web sont en forme de ligne. **Line** : les liens entre les schémas et les paramètres du service Web sont en forme de lignes droites. Cette option vous permettra d'optimiser les performances.
	Property type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données
	Port	Numéro de port d'écoute du serveur de la base.
	Database	Nom de la base de données
	Username et *Password*	Données d'authentification de l'utilisateur de la base
	Encoding Type	Sélectionnez l'encodage dans la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour le traitement des données d'une base de données.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	L'utilisation du composant **tELTMSSqlMap** est étroitement liée à celle des composants **tELTMSSqlInput** et **tELTMSSqlOutput**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénarios associés

Pour des scénarios associés, consultez les scénarios suivants du composant **tELTMysqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTMSSqlOutput

Propriétés de tELTMSSqlOutput

Les composants **tELTMSSqlInput**, **tELTMSSqlOutput**, et **tELTMSSqlMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base MSSql afin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille du composant	ELT	
Fonction	Effectue l'action sur la table telle que spécifiée et insère les données selon le schéma de sortie défini dans le Mapper ELT.	
Objectif	Exécute la requête Insert dans la base de données MSSql.	
Basic settings	*Action on data*	Sur les données de la table définie, vous pouvez effectuer les opérations suivantes : **Insert** : Ajoute de nouvelles lignes à la table. Si des doublons sont identifiés, le Job s'arrête. Note : dans les composants ELT MSSql, seule l'opération Insert est disponible.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Default Table Name	Saisissez le nom de la table par défaut, entre guillemets doubles.
	Default Schema Name	Saisissez le nom du schéma par défaut, entre guillemets doubles.
	Use different table name	Cochez cette case afin de définir le nom d'une table de sortie différente, entre guillemets doubles, dans le champ **Table name** qui apparaît.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	L'utilisation du composant **tELTMSSqlOutput** est étroitement liée à celle des composants **tELTMSSqlInput** et **tELTMSSqlMap**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.

Scénarios associés

Pour des scénarios utilisant un composant semblable au **tELTMSSqlOutput**, consultez les scénarios suivants du composant **tELTMysqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTMysqlInput

Propriétés de tELTMysqlInput

Les composants **tELTMysqlInput**, **tELTMysqlOutput**, et **tELTMysqlMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base MySQL afin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille de composant	ELT	
Fonction	Fournit le schéma de la table d'entrée à la commande SQL qui sera exécutée.	
Objectif	Permet d'ajouter autant de tables que nécessaires dans une commande Insert qui peut être complexe.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités. Le schéma est soit local (built-in) soit distant dans le Repository. Le schéma défini est ensuite passé au composant ELT Mapper afin de l'utiliser dans la commande SQL Insert Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Default Table Name	Saisissez le nom de la table par défaut, entre guillemets doubles.
Utilisation	L'utilisation du composant **tELTMysqlInput** est étroitement liée à celle du **tELTMysqlMap**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénarios associés

Pour des scénarios utilisant le composant **tELTMysqlInput**, consultez les scénarios suivants du composant **tELTMysqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTMysqlMap

Propriétés de tELTMysqlMap

Les composants **tELTMysqlInput**, **tELTMysqlOutput**, et **tELTMysqlMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base MySQL afin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille de composant	ELT	
Fonction	Permet de construire votre commande SQL graphiquement en utilisant la table fournie en entrée.	
Objectif	Utilise les tables fournies en entrée pour alimenter les paramètres nécessaires à l'exécution de la commande SQL souhaitée. La commande peut contenir des jointures internes comme externes entre différentes tables ou entre une table et ses alias.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données
	Port	Numéro de port d'écoute du serveur de la base.
	Database	Nom de la base de données
	Username et *Password*	Données d'authentification de l'utilisateur de la base
	Encoding	Sélectionnez l'encodage dans la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour le traitement des données d'une base de données.
	Preview	L'aperçu est un cliché des données traitées dans le Mapper. Il apparaît lorsque les propriétés du Mapper ont été complétées. La synchronisation de l'aperçu n'est disponible qu'après avoir enregistrer les modifications.
	Map editor	L'éditeur ELT du Mapper vous permet de définir le schéma de sortie ainsi que de construire graphiquement la commande SQL à exécuter.
Utilisation	L'utilisation du composant **tELTMysqlMap** est étroitement liée à celle des composants **tELTMysqlInput** et **tELTMysqlOutput**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Connecter les composants ELT

Les composants ELT ne traitent pas les données en tant que telles mais les informations du schéma descriptif de la table qui seront utilisées pour alimenter les paramètres de la requête SQL à exécuter.

Par conséquent la connexion requise pour relier ces différents composants est simplement un lien qui porte le nom de la table.

Le nom de la sortie que vous donnez à ce lien lors de sa création doit correspondre exactement au nom de la table car ce paramètre sera utilisé lors de la génération de la commande SQL.

Voir également : *Connexion de type Link* du **Guide Utilisateur** **Talend Open Studio**

Tables de jointure et de correspondance

Dans le Mapper ELT, vous pouvez sélectionner des colonnes spécifiques des schémas d'entrée et les ajouter dans le schéma de sortie.

- De la même manière que dans l'éditeur du Mapper ETL, simplement glissez et déposez le contenu du schéma d'entrée vers le schéma de sortie.
- Utilisez les touches Ctrl et Maj pour la sélection multiple des colonnes contiguës ou non.

Vous pouvez mettre en place des jointures explicites pour extraire des données variées de différentes tables.

- Cliquez sur la liste déroulante **Join** et sélectionnez une jointure explicite.
- Les jointures suivantes sont disponibles : **Inner Join**, **Left Outer Join**, **Right Outer Join** ou **Full Outer Join** et **Cross Join**.
- Par défaut la jointure interne **(Inner Join)** est sélectionnée.

Vous pouvez également créer des tables **Alias** afin de sélectionner des données variées dans une même table.

- Dans la zone Input, cliquez sur le bouton plus (+) pour créer un Alias.
- Définissez la table sur laquelle l'alias est basé.
- Saisissez un nom pour la table alias, de préférence veillez à ce qu'il soit différent de la table principale pour éviter les confusions.

Ajouter des clauses Where

Vous pouvez également restreindre la requête Select à l'aide d'une clause Where. Cliquez sur le bouton **Add filter row** situé en haut de la table de sortie et saisissez la condition que vous souhaitez appliquer.

Veillez à ce que les composants d'entrée soient tous liés correctement au composant ELT Map pour garantir que les inclusions, jointures et clauses fonctionnent.

Générer la requête SQL

La mise en correspondance des éléments à partir des schémas d'entrée vers les schémas de sortie crée automatiquement la requête Select correspondante.

Schema editor | Expression editor | Generated SQL Select query for 'table1' output

```
SELECT
owners.ID_Owner, owners.Name_Customer, owners.ID_Insurance, cars.Reg_Car, cars.Make, cars.Color,
 resellers.Name_Reseller, resellers.City
FROM
 owners INNER JOIN  cars ON(  cars.ID_Owners = owners.ID_Owner )
 INNER JOIN  resellers ON(  resellers.ID_Reseller = cars.ID_Reseller )
WHERE resellers.City ='West Coast City'
```

Les éventuelles clauses sont également prises en compte automatiquement.

Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre

Le scénario suivant décrit un Job rassemblant plusieurs schémas d'une table en entrée puis utilise une clause dans la requête SQL afin de filtrer les données en sortie.

- Cliquez et déposez les composants suivants : **tELTMysqlIntput**, **tELTMysqlMap**, **tELTMysqlOutput**.
- Trois composants sont requis en entrée pour ce Job.
- Connectez les trois composants ELT d'entrée au mapper ELT à l'aide de liens dont le nom doit suivre strictement la nomenclature des tables de la base de données, dans cet exemple : *owners*, *cars* et *resellers*.
- Puis connectez le mapper ELT au composant de sortie ELT à l'aide d'un nouveau lien appelé *results* dans cet exemple.
- Les trois schémas d'entrée sont stockés dans les métadonnées (**Metadata**) du **Repository**. Ils peuvent donc facilement être réutilisés.

- Cliquez sur le mapper ELT pour définir les informations de connexion à la base de données.
- Les informations de connexion à la base de données sont là encore stockées dans le Repository.

- L'encodage par défaut est utilisé pour la base de données MySql.
- Ouvrez l'éditeur ELT Map pour mettre en place les jointures entre les tables d'entrée (Input).
- Glissez-déposez la colonne *ID_Owner* à partir de la table *Owners* vers la colonne correspondante dans la table *cars*.
- Sélectionnez l'option de jointure **INNER JOIN** dans la table *Cars*, puis cochez la case **Explicit Join** en face de *ID_Owners*.
- Glissez la colonne *ID_Resellers* à partir de la table *Cars* vers la table *Resellers* pour mettre en place la seconde jointure. Sélectionnez à nouveau l'option de jointure **INNER JOIN** dans la table *Resellers* et cochez la case **Explicit Join** face à la colonne correspondante.
- Puis sélectionnez les colonnes à agréger en sortie.
- Sélectionnez toutes les colonnes des tables *Cars* et *Owners* et uniquement les colonnes *Reseller_Name* et *City* à partir de la table *Resellers*.
- Glissez et déposez ces colonnes dans la table de sortie *Results*.
- La mise en correspondance apparaît en jaune et les jointures en violet.

- Cliquez sur l'onglet **Generated SQL Select query** pour afficher la requête SQL correspondante.

- Puis appliquez un filtre sur la table de sortie.
- Cliquez sur le bouton **Add filter row** de la table de sortie (output).

- Conditionnez le Select à l'aide d'une clause Where comme suit : resellers.City ='West Coast City'
- Notez que la clause Where apparaît dans l'onglet d'aperçu de la requête SQL générée.

- Cliquez **OK** pour enregistrer le paramétrage de l'ELT Map.
- Configurez le composant de sortie ELT Output dans la vue Component du composant **tELTMysqlOutput**.
- Dans le champ **Action on table** sélectionnez **Drop and create table** pour l'exemple donné ci-dessous et dans ce cas, la seule action disponible sur les données est l'insertion, donc sélectionnez **Insert**.
- Le schéma doit être synchronisé avec le composant **tELTMysqlMap** pour que les divers schémas source puissent être agrégés.

Resultset 1

ID...	Name_Customer	ID_Insur...	Reg_Car	Make	Color	Name_Reseller	City
4	hirtken	ENX9366	6225 GT 57	Renault	green	Best Cars Shop	West Coast City
16	kennan	RTA8580	0601 SQ 67	Renault	blue	Cars & Pickup Re...	West Coast City
31	antken	QDG0199	5729 DJ 52	Renault	blue	Cars & Pickup Sp...	West Coast City
33	hirtken	MYA1613	0427 LR 72	Toyota	gold	Best Cars Specialist	West Coast City
34	nanant	XBM2459	7355 IB 28	BMW	purple	Best Cars Resale	West Coast City
39	nanneng	CSP8847	8402 JE 03	Volkswa...	blue	Cars & Pickup Re...	West Coast City
62	gallken	LZP1021	3940 ZW 19	BMW	green	All you need Outlet	West Coast City
65	oinele	ANX9956	6523 DY 26	Mercedes	gold	Best Cars Shop	West Coast City
76	nengle	SLG5853	2087 IW 01	Toyota	gold	Best Cars Outlet	West Coast City
77	boot	QFS6844	3795 GN 95	Toyota	blue	All you need Outlet	West Coast City
88	carbo	EHN3338	7752 OB 89	Mercedes	blue	Cars & Pickup Re...	West Coast City
89	bobouh	UMT0348	0462 FV 53	Toyota	purple	Best Cars Outlet	West Coast City

Toutes les données sélectionnées sont insérées dans la table *results* comme spécifié dans la requête SQL statement conditionnée par la clause.

Scénario 2 : ELT utilisant une table Alias

Le scénario suivant décrit un Job utilisant une table Alias. La table *employees* contient toutes les informations concernant les employées ainsi qu'un ID de leur responsable respectif, qui sont également considérés comme des employés et par conséquent inclus dans la table *employees*. La table *dept* contient les informations de lieu et de département des employés de la base.

- Glissez et déposez les composants **tELTMysqlInput** afin de collecter les schémas respectifs des tables *employees* et *dept*.
- Dans cet exemple, les deux schémas sont stockés dans le **Repository** et peuvent ainsi être facilement réutilisés.

- Puis sélectionnez le composant **tELTMysqlMap** et configurez les informations de connexion à la base Mysql.

- Ces informations de connexion sont là encore stockées dans la partie Metadata du Repository.

- Cliquez sur le bouton **[...]** pour lancer l'éditeur ELT Map.
- Vérifiez d'abord que la première table d'entrée de la zone Input correspond à votre flux principal car la réussite des jointures en dépend.
- Dans cet exemple, la table *employees* devrait donc se trouver en première position.

- Glissez-déposez la colonne *DeptNo* à partir de la table *employees* vers la table *dept* afin de mettre en place la jointure **join** entre les deux tables d'entrée.
- Cochez la case **Explicit Join** et définissez cette jointure comme **Inner Join**.
- Puis créez la table **Alias** basée sur la table *employees*.

- Appelez cet alias *Managers* et cliquez OK pour l'afficher comme nouvelle table d'entrée (Input) dans le mapper ELT.
- Glissez-déposez la colonne *ID* column à partir de la table *employees* vers la colonne *ID_Manager* de la table alias *Managers* nouvellement créée.
- Cochez la case **Explicit Join** et définissez la comme **Left Outer Join**, afin que les résultats de ce filtre soient récupérés même en cas de valeur **Null**.

- Glissez-déposez le contenu des deux tables d'entrée, *employees* et *dept*, ainsi que la colonne *Name* à partir de la table *Manager* vers la table de sortie (Output).
- Cliquez sur l'onglet **Generated SQL Select query** pour afficher la requête à exécuter.

Schema editor | Expression editor | Generated SQL Select query for 'table2' output

```
SELECT
employees.ID, employees.DEPTNO, employees.NAME, employees.ID_MANAGER, dept.DNAME, dept.LOC,
 Managers.NAME
FROM
 employees INNER JOIN  dept ON(  dept.DEPTNO = employees.DEPTNO )
 LEFT OUTER JOIN  employees Managers ON(  Managers.ID_MANAGER = employees.ID  )
```

- Puis cliquez sur le composant de sortie et configurez le champ **Action on data** sur **Insert**.
- Assurez-vous que le schéma est synchronisé avec la table de sortie du mapper ELT avant d'exécuter le Job via la touche **F6** ou via la barre d'outils.

Resultset 1

ID	DEPTNO	NAME	ID_MANAGER	DNAME	LOC	NAME_1
1	10	AXEL	6	ACCOUNTING	NEW YORK	STEPHANE
2	10	PIERRICK	6	ACCOUNTING	NEW YORK	NULL
3	20	MICHAEL	6	RESEARCH	DALLAS	NULL
4	10	STEPHANE	1	ACCOUNTING	NEW YORK	NULL
6	10	CEDRIC	7	ACCOUNTING	NEW YORK	AXEL
6	10	CEDRIC	7	ACCOUNTING	NEW YORK	PIERRICK
6	10	CEDRIC	7	ACCOUNTING	NEW YORK	MICHAEL
6	10	CEDRIC	7	ACCOUNTING	NEW YORK	CHRISTOPHE
7	10	FABRICE	NULL	ACCOUNTING	NEW YORK	CEDRIC

L'information *Department* ainsi que les entrées *Employees* sont associées dans la sortie, et la colonne *Manager Name* a pu être récupérée grâce à la jointure explicite.

tELTMysqlOutput

Propriétés de tELTMysqlOutput

Les composants **tELTMysqlInput**, **tELTMysqlOutput**, et **tELTMysqlMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base MySQL afin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille du composant	ELT	
Fonction	Effectue l'action sur la table telle que spécifiée et insère les données selon le schéma de sortie défini dans le Mapper ELT.	
Objectif	Exécute la requête Insert dans la base de données Mysql	
Basic settings	*Action on data*	Sur les données de la table définie, vous pouvez effectuer les opérations suivantes : **Insert** : Ajoute de nouvelles lignes à la table. Si des doublons sont identifiés, le Job s'arrête. Note : dans les composants ELT Mysql, seule l'opération Insert est disponible.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Default Table Name	Saisissez le nom de la table par défaut, entre guillemets doubles.
	Use different table name	Cochez cette case afin de définir le nom d'une table de sortie différente, entre guillemets doubles, dans le champ **Table name** qui apparaît.
Utilisation	L'utilisation du composant **tELTMysqlOutput** est étroitement liée à celle des composants tELTMysqlInput et **tELTMysqlMap**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénarios associés

Pour des scénarios utilisant le composant **tELTMysqlOutput**, consultez les scénarios suivants du composant **tELTMysqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTOracleInput

Propriétés de tELTOracleInput

Les composants **tELTOracleInput**, **tELTOracleOutput**, et **tELTOracleMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base Oracle afin de générer des commandes Insert, Udpate ou Delete y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille de composant	ELT	
Fonction	Fournit le schéma de la table d'entrée à la commande SQL qui sera exécutée.	
Objectif	Permet d'ajouter autant de tables que nécessaires dans une commande Insert qui peut être complexe.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités. Le schéma est soit local (built-in) soit distant dans le Repository. Le schéma défini est ensuite passé au composant ELT Mapper afin de l'utiliser dans la commande SQL Insert. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Default Table Name	Saisissez le nom de la table par défaut, entre guillemets doubles.
Java uniquement	*Default Schema Name*	Saisissez le nom du schéma par défaut, entre guillemets doubles.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	L'utilisation du composant **tELTOracleInput** est étroitement liée à celle du **tELTOracleMap**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénario associé

Pour un scénario utilisant le composant **tELTOracleInput**, consultez le *Scénario : Mettre à jour les lignes d'une base Oracle, page 853* du composant **tELTOracleMap**.

tELTOracleMap

Propriétés de tELTOracleMap

Les composants **tELTOracleInput**, **tELTOracleOutput**, et **tELTOracleMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base Oracle afin de générer des commandes Insert, Update ou Delete y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille de composant	ELT	
Fonction	Permet de construire votre commande SQL graphiquement en utilisant la table fournie en entrée.	
Objectif	Utilise les tables fournies en entrée pour alimenter les paramètres nécessaires à l'exécution de la commande SQL souhaitée. La commande peut contenir des jointures internes comme externes entre différentes tables ou entre une table et ses alias.	
Basic settings	*Use an existing connection* *Java uniquement*	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	ELT Oracle Map Editor	L'éditeur ELT du Mapper vous permet de définir le schéma de sortie ainsi que de construire graphiquement la commande SQL à exécuter.

	Style link	Sélectionnez le type d'affichage des liens. **Auto** : par défaut, les liens entre les schémas d'entrée et de sortie et les paramètres du service Web sont en forme de courbe. **Curves** : les liens entre les schémas et les paramètres du service Web sont en forme de ligne. **Line** : les liens entre les schémas et les paramètres du service Web sont en forme de lignes droites. Cette option vous permettra d'optimiser les performances.
	Property type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
Java uniquement	*Connection type*	Liste déroulante des pilotes disponibles.
	Host	Adresse IP du serveur de base de données
	Port	Numéro de port d'écoute du serveur de la base.
	Database	Nom de la base de données
	Username et *Password*	Données d'authentification de l'utilisateur de la base
	Encoding	Sélectionnez l'encodage dans la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour le traitement des données d'une base de données.
Advanced settings	*Additional JDBC Parameters* *Java uniquement*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
Java uniquement	*Use Hint Options*	Cochez cette case afin d'activer la zone de configuration des hints, afin de vous permettre d'optimiser l'exécution d'une requête. Dans cette zone, les paramètres sont : -**HINT** : spécifiez le hint dont vous avez besoin, à l'aide de la syntaxe `/*+ */`. -**POSITION** : spécifiez où placer le hint dans l'instruction SQL. -**SQL STMT** : sélectionnez l'instruction SQL à utiliser.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	L'utilisation du composant **tELTOracleMap** est étroitement liée à celle des composants tELTOracleInput et **tELTOracleOutput**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Connecter les composants ELT

Pour plus d'informations concernant la connexion des composants ELT, consultez *Connecter les composants ELT, page 839.*

Voir également : *Connexion de type Link* du **Guide Utilisateur** **Talend Open Studio**

Tables de jointure et de correspondance

Dans le Mapper ELT, vous pouvez sélectionner des colonnes des schémas d'entrée et les intégrer dans le schéma de sortie.

Pour plus d'informations concernant la mise en correspondance et les jointures, consultez *Tables de jointure et de correspondance, page 839.*

Ajouter des clauses Where

Pour plus d'informations sur la manipulation des clauses, consultez *Ajouter des clauses Where, page 839.*

Générer la requête SQL

La mise en correspondance d'éléments des schémas d'entrée avec le schéma de sortie crée automatiquement la requête Select correspondante.

La/les clauses définies dans le mapper ELT sont également intégrées automatiquement dans la requête.

Scénario : Mettre à jour les lignes d'une base Oracle

Le scénario suivant est basé sur le Job décrit dans la section des composants ELT MySQL, *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840*. L'action sur les données Update étant disponible en base Oracle, le scénario suivant décrit un Job qui met à jour une sélection de lignes provenant de la table *results*, puis ajoute l'information *model* (modèle) à la colonne *make* (marque) de la table *cars* (véhicules).

- Configurez les trois composants d'entrée tel que décrit dans *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840*.
- Lors de cette étape de connexion des composants d'entrée ELT, assurez-vous que le nom des liens correspond exactement aux noms des tables de la base de données car ils seront utilisés comme paramètres dans la requête SQL générée grâce au mapper ELT.
- Enlevez la clause utilisée pour filtrer les colonnes de sortie (si vous partez du scénario MySql indiqué en introduction).
- Ajoutez une nouvelle ligne de filtre à la table de sortie pour établir une relation entre tables d'entrée (*owners*) et de sortie (*results*) : owners.ID_OWNER=results.ID_OWNER

- Puisque notre but est de mettre à jour la table, enlevez toutes les colonnes de la table de sortie qui ne sont pas concernées par cette mise à jour (**Update**).
- Puis mettez à jour la colonne *Make* (marque) en ajoutant la mention *C-Class* précédée d'une double barre verticale.
- Ajoutez également la mention *Sold by* (Vendu par) devant le nom du revendeur dans la colonne *reseller name* (nom revendeur) de la table *resellers*.
- Vérifiez la requête Select générée dans l'onglet **Generated SQL select query**.

- Cliquez **OK** pour valider la configuration et fermer le mapper ELT. Veillez à ce que les informations de connexion à la base Oracle soient correctes dans l'onglet **Basic Settings** du composant **tELTOracleMap**.
- Puis sélectionnez le composant **tELTOracleOutput** afin de définir les actions à effectuer sur les données dans le champ **Action on data**.

- Aucune action n'est requise sur la table, mais par contre sélectionnez **Update** dans le champ **Action on data**.
- Cliquez **Edit schema** pour vérifier que le schéma correspond à celui de la table de sortie configuré dans le Mapper ELT.
- Dans le champ **Where clause**, entrez la clause supplémentaire suivante : `results.MAKE= 'Mercedes'`.
- Puis appuyez sur **F6** pour exécuter le Job puis vérifiez la table *results* dans un client de base de données.

```
Starting job ELTOracleUpdate at 12:52 11/04/2007.
Updating with :
UPDATE results SET (MAKE,NAME_RESELLER) = (SELECT cars.MAKE || '
C-Class', cars.REG_CAR || ' sold by ' || resellers.NAME_RESELLER
FROM  owners , cars , resellers WHERE    cars.ID_OWNERS =
owners.ID_OWNER   AND  resellers.ID_RESELLER = cars.ID_RESELLER
AND owners.ID_OWNER=results.ID_OWNER)   WHERE results.MAKE=
'Mercedes'

--> 2 rows updated.

Job ELTOracleUpdate ended at 12:52 11/04/2007. [exit code=0]
```

Le Job a donc exécuté la requête SQL générée et effectué les mises à jour des lignes concernées.

BMW	green	98	All you need Outlet	West Coast City
Mercedes C-Class	gold	3	6523 DY 26 sold by Best Cars Shop	West Coast City
Toyota	gold	84	Best Cars Outlet	West Coast City
Toyota	blue	98	All you need Outlet	West Coast City
Mercedes C-Class	blue	58	7752 OB 89 sold by Cars & Pickup Resale	West Coast City

tELTOracleOutput

Propriétés de tELTOracleOutput

Les composants **tELTOracleInput**, **tELTOracleOutput**, et **tELTOracleMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base Oracle afin de générer des commandes Insert, Update ou Delete y compris des clauses, et ces commandes devront être exécutées dans la table de sortie définie.

Famille du composant	ELT	
Fonction	Effectue l'action sur la table telle que spécifiée et insère les données selon le schéma de sortie défini dans le Mapper ELT.	
Objectif	Exécute la requête Insert ou Update dans la base de données Oracle	
Basic settings *L'option MERGE est disponible en Java uniquement*	*Action on data*	Sur les données de la table définie, vous pouvez effectuer les opérations suivantes : **Insert** : Ajoute de nouvelles lignes à la table. Si des doublons sont identifiés, le Job s'arrête. **Update** : Met à jour des lignes existantes. **Delete** : Supprime les entrées correspondantes au flux d'entrée. **MERGE** : met à jour ou insère les données dans la table. Les options disponibles pour l'opération MERGE diffèrent de celles disponibles pour les opérations Insert, Update et Delete.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Where clauses (for UPDATE and DELETE only)	Saisissez la clause permettant de filtrer les données à mettre à jour ou à supprimer lors des opérations de mise à jour ou de suppression.

Java uniquement	*Use Merge Update* (pour **MERGE**)	Cochez cette case pour mettre à jour des données de votre table de sortie. **Column** : Liste des colonnes du flux d'entrée. **Update** : Cochez la case correspondant au nom de la colonne que vous souhaitez mettre à jour. **Use Merge Update Where Clause** : Cochez cette case et saisissez la clause WHERE permettant de filtrer les données à mettre à jour, si nécessaire. **Use Merge Update Delete Clause** : Cochez cette case et saisissez la clause WHERE permettant de filtrer les données à supprimer et mettre à jour, si nécessaire.
Java uniquement	*Use Merge Insert* (pour **MERGE**)	Cochez cette case pour insérer des données dans la table. **Column** : Liste des colonnes du flux d'entrée. **Check All** : Cochez la case correspondant au nom de la colonne que vous souhaitez insérer. **Use Merge Update Where Clause** : Cochez cette case et saisissez la clause WHERE permettant de filtrer les données à insérer.
	Default Table Name	Saisissez le nom de la table par défaut.
Java uniquement	*Default Schema Name*	Saisissez le nom du schéma Oracle par défaut.
	Use different table name	Cochez cette case afin de définir le nom d'une table de sortie différente, entre guillemets doubles, dans le champ **Table name** qui apparaît.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	L'utilisation du composant **tELTOracleOutput** est étroitement liée à celle des composants **tELTOracleInput** et **tELTOracleMap**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénario : Utiliser la fonction MERGE d'Oracle pour mettre à jour et insérer simultanément des données

Le scénario suivant décrit un Job à trois composants ELT permettant de mettre à jour les données d'une table client grâce à la commande MERGE d'Oracle. Les informations des nouveaux clients sont ajoutées et celles des anciens sont mises à jour.

A partir du noeud **Metadata** du **Repository** :

- Sous le noeud correspondant à la base de données contenant les tables à utiliser pour ce scénario, sélectionnez le schéma de la table contenant les nouvelles données sur les clients et glissez dans l'espace de modélisation. Dans ce scénario, le schéma utilisé se nomme *NEW_CUSTOMERS_MLE*.
- Dans la boîte de dialogue **[Components]**, sélectionnez le composant **tELTOracleInput** et cliquez sur **OK**.

Pour plus d'informations concernant les métadonnées, consultez la section *Centraliser les métadonnées* du **Guide Utilisateur** de **Talend Open Studio**.

A partir de la **Palette** de composants :

- Glissez les composants **tELTOracleMap** et **tELTOracleOutput** dans l'espace de modélisation.

Pour relier les composants entre eux :

- Cliquez-droit sur le composant d'entrée, *NEW_CUSTOMERS_MLE*, glissez jusqu'au **tELTOracleMap** et relâchez le bouton droit pour créer le lien.
- De la même manière, connectez le **tELTOracleMap** au **tELTOracleOutput** à l'aide d'un nouveau lien appelé *RESULTS* dans cet exemple.
- Double-cliquez sur le composant d'entrée pour afficher ses propriétés dans la vue **Components**.

- Dans la liste **Schema**, sélectionnez **Repository** si vous avez stocké le schéma de votre table dans le répertoire **Metadata** du **Repository** ou sélectionnez **Built-in** et configurez-le manuellement en cliquant sur le bouton **[...]** à côté du champ **Edit schema**.

Le schéma est configuré comme suit :

- Dans le champ **Default Table Name**, saisissez entre guillemets le nom de la table source. Ici, la table utilisée se nomme *NEW_CUSTOMERS_MLE*.
- Dans le champ **Default Schema Name**, saisissez entre guillemets le nom du schéma source. Ici, le schéma utilisé se nomme *ROOT*.
- Cliquez sur le **tELTOracleMap** et sur la vue **Component** pour afficher les propriétés de base du composant.

- Cochez la case **Use an existing connection** et sélectionnez le composant **tOracleConnection** dans la liste **Component list** si vous souhaitez utiliser la connexion ouverte par un composant de connexion.
- Dans la liste **Property Type**, sélectionnez **Repository** et cliquez sur le bouton **[...]** pour sélectionner la connexion à la base de données contenant les tables utilisées dans ce scénario. Les champs suivants seront automatiquement renseignés avec les informations de connexion à cette base de données. Si vous n'avez pas créé de métadonnées avec ces informations, sélectionnez le mode **Built-in** dans la liste **Property Type** et renseignez-les manuellement.

Pour plus d'informations concernant les métadonnées, consultez la section *Centraliser les métadonnées* du **Guide Utilisateur** de Talend Open Studio.

- Cliquez sur le bouton **[...]** correspondant au champ **ELT Oracle Map Editor** pour ouvrir l'éditeur permettant de faire les correspondances entre le flux d'entrée et le flux de sortie.

- Cliquez sur le bouton **[+]** en haut à gauche de l'éditeur pour ouvrir la boîte de dialogue permettant d'ajouter l'alias correspondant à la table d'entrée.

Dans la boîte de dialogue **Add a new alias** :

- Sélectionnez le nom du flux de la table d'entrée dans la liste **Select the table to use**. Ici, vous avez sélectionné *ROOT.NEW_CUSTOMERS_MLE*.
- Saisissez le nom que vous souhaitez donner à l'alias correspondant dans le champ **Type in a valid alias**. Ici, vous avez nommé l'alias *new_customers*.

  L'alias n'est pas obligatoire quand la table en entrée n'est utilisée qu'une seule fois dans le **tMap**.

- Cliquez sur **OK**.
- Dans l'éditeur du **tELTOracleMap**, sélectionnez toutes les colonnes de la table d'entrée et glissez-les dans la table de sortie, *RESULTS*.
- Cliquez sur l'onglet **Generated SQL Select query** pour afficher la requête à exécuter.

- Cliquez sur **OK**.
- Dans l'espace de modélisation, double-cliquez sur le composant **tELTOracleOutput** pour paramétrer ses propriétés dans la vue **Component**.

- Dans la liste **Action on data**, sélectionnez **MERGE**.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent ou paramétrez manuellement le schéma de sortie en sélectionnant **Built-in** dans la liste **Schema** et en cliquant sur le bouton **[...]** à côté du champ **Edit schema**.
- Cochez la case **Use Merge Update** pour mettre à jour les données grâce à la fonction MERGE d'Oracle.

- Dans le tableau qui apparaît, cochez les cases correspondant aux colonnes que vous souhaitez mettre à jour. L'objectif est de mettre à jour les données client en fonction de leur ID. Ainsi, cochez toutes les cases sauf celle correspondant à la colonne ID.

Les colonnes définies comme clé primaire NE PEUVENT PAS et NE DOIVENT PAS faire l'objet d'une mise à jour.

- Cochez les cases **Use Merge Update Where Clause** et **Use Merge Update Delete Clause** si vous souhaitez utiliser une clause WHERE pour filtrer les données à mettre à jour et une clause DELETE WHERE pour supprimer les données avant de réinsérer les données mises à jour, lors de l'opération de MERGE.

- Cochez la case **Use Merge Insert** pour insérer de nouvelles données en même temps que de mettre à jour les anciennes grâce à la fonction MERGE d'Oracle.
- Dans le tableau qui apparaît, cochez les cases correspondant aux colonnes que vous souhaitez insérer. L'objectif est d'insérer toutes les nouvelles données client. Ainsi, cochez toutes les cases correspondant de toutes les colonnes à insérer.
- Cochez la case **Use Merge Insert Where Clause** si vous souhaitez utiliser une clause WHERE pour filtrer les données à insérer lors de l'opération de MERGE.

- Dans le champ **Default Table Name**, saisissez entre guillemets le nom de la table source. Ici, la table utilisée se nomme *NEW_CUSTOMERS_MLE*
- Dans le champ **Default Schema Name**, saisissez entre guillemets le nom du schéma source. Ici, le schéma utilisé se nomme *ROOT*.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job tELTOracleOutput_MERGE at 18:23 23/11/2009.

[statistics] connecting to socket on port 4185
[statistics] connected
Merge with :
MERGE INTO ROOT.RESULTS target USING (SELECT new_customers.ID ,
new_customers.FIRSTNAME, new_customers.LASTNAME, new_customers.ADDRESS,
new_customers.REGISTERDATE, new_customers.REVENUE, new_customers.STATES FROM
ROOT.NEW_CUSTOMERS_MLE new_customers) source ON (target.ID=source.ID) WHEN
MATCHED THEN UPDATE SET
target.FIRSTNAME=source.FIRSTNAME,target.LASTNAME=source.LASTNAME,target.ADD
RESS=source.ADDRESS,target.REGISTERDATE=source.REGISTERDATE,target.REVENUE=s
ource.REVENUE,target.STATES=source.STATES  WHEN NOT MATCHED THEN INSERT (
ID,FIRSTNAME,LASTNAME,ADDRESS,REGISTERDATE,REVENUE,STATES) VALUES (
source.ID,source.FIRSTNAME,source.LASTNAME,source.ADDRESS,source.REGISTERDAT
E,source.REVENUE,source.STATES)
[statistics] disconnected
Job tELTOracleOutput_MERGE ended at 18:23 23/11/2009. [exit code=0]
```

Les données sont mises à jour et insérées dans la base de données et la requête utilisée s'affiche dans la console.

Pour un scénario utilisant le composant **tELTOracleOutput**, consultez le *Scénario : Mettre à jour les lignes d'une base Oracle, page 853* du composant **tELTOracleMap**.

tELTPostgresqlInput

Propriétés de tELTPostgresqlInput

Les composants **tELTPostgresqlInput**, **tELTPostgresqlOutput**, et **tELTPostgresqlMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base Postgresql afin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille de composant	ELT	
Fonction	Fournit le schéma de la table d'entrée à la commande SQL qui sera exécutée.	
Objectif	Permet d'ajouter autant de tables que nécessaires dans une commande Insert qui peut être complexe.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités. Le schéma est soit local (built-in) soit distant dans le Repository. Le schéma défini est ensuite passé au composant ELT Mapper afin de l'utiliser dans la commande SQL Insert Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Default Table Name	Saisissez le nom de la table par défaut.
	Default Schema Name	Saisissez le nom du schéma par défaut.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	L'utilisation du composant **tELTPostgresqlInput** est étroitement liée à celle du **tELTPostrgesqlMap**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénarios associés

Pour des scénarios utilisant un composant semblable au **tELTPostgresqlInput**, consultez les scénarios suivants du composant **tELTPostgresqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTPostgresqlMap

Propriétés de tELTPostgresqlMap

Les composants **tELTPostgresqlInput**, **tELTPostgresqlOutput**, et **tELTPostgresqlMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base Postgresql afin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille de composant	ELT	
Fonction	Permet de construire votre commande SQL graphiquement en utilisant la table fournie en entrée.	
Objectif	Utilise les tables fournies en entrée pour alimenter les paramètres nécessaires à l'exécution de la commande SQL souhaitée. La commande peut contenir des jointures internes comme externes entre différentes tables ou entre une table et ses alias.	
Basic settings	*Use an existing connection*	Cochez cette case et sélectionnez le composant **tPostgresqlConnection** adéquat à partir de la liste **Component** pour réutiliser les paramètres d'une connexion que vous avez déjà définie. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	ELT Postgresql Map Editor	L'éditeur ELT du Mapper vous permet de définir le schéma de sortie ainsi que de construire graphiquement la commande SQL à exécuter.

	Style link	Sélectionnez le type d'affichage des liens. **Auto** : par défaut, les liens entre les schémas d'entrée et de sortie et les paramètres du service Web sont en forme de courbe. **Curves** : les liens entre les schémas et les paramètres du service Web sont en forme de ligne. **Line** : les liens entre les schémas et les paramètres du service Web sont en forme de lignes droites. Cette option vous permettra d'optimiser les performances.
	Property type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données
	Port	Numéro de port d'écoute du serveur de la base.
	Database	Nom de la base de données
	Username et *Password*	Données d'authentification de l'utilisateur de la base
	Encoding Type	Sélectionnez l'encodage dans la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour le traitement des données d'une base de données.
Advanced settings	*Additional JDBC parameters*	Spécifiez des informations supplémentaires de connexion à la base de données créée. Cette option n'est pas disponible lorsque vous utilisez l'option **Use an existing connection** dans les **Basic settings**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	L'utilisation du composant **tELTPostgresqlMap** est étroitement liée à celle des composants **tELTPostgresqlInput** et **tELTPostgresqlOutput**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénarios associés

Pour des scénarios associés, consultez les scénarios suivants du composant **tELTMysqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTPostgresqlOutput

Propriétés de tELTPostgresqlOutput

Les composants **tELTPostgresqlInput**, **tELTPostgresqlOutput**, et **tELTPostgresqlMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base Postgresql afin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille du composant	ELT	
Fonction	Effectue l'action sur la table telle que spécifiée et insère les données selon le schéma de sortie défini dans le Mapper ELT.	
Objectif	Exécute la requête Insert dans la base de données Postgresql.	
Basic settings	*Action on data*	Sur les données de la table définie, vous pouvez effectuer les opérations suivantes : **Insert** : Ajoute de nouvelles lignes à la table. Si des doublons sont identifiés, le Job s'arrête. Note : dans les composants ELT Postrgesql, seule l'opération Insert est disponible.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Default Table Name	Saisissez le nom de la table par défaut.
	Default Schema Name	Saisissez le nom du schéma par défaut.
	Use different table name	Cochez cette case afin de définir le nom d'une table de sortie différente, entre guillemets doubles, dans le champ **Table name** qui apparaît.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	L'utilisation du composant **tELTPostgresqlOutput** est étroitement liée à celle des composants **tELTPostgresqlInput** et **tELTPostgresqlMap**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénarios associés

Pour des scénarios utilisant un composant semblable au **tELTPostgresqlOutput**, consultez les scénarios suivants du composant **tELTMysqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTRollback

Propriétés du tELTRollback

Ce composant est étroitement lié au composant **tELTCommit** et au composant ELT de connexion correspondant à la base de données sur laquelle vous travaillez. Ils sont généralement utilisés ensemble lors de transactions.

Famille du composant	Databases/ELT	
Fonction	Le **tELTRollback** annule la transaction dans une base de données connectée.	
Objectif	Ce composant évite le commit de transaction involontaire.	
Basic settings	*Database Type*	Sélectionnez dans le menu déroulant le type de base de données sur lequel vous voulez travailler.
	Component list	Sélectionnez le composant ELT de connexion le plus pertinent dans la liste si vous prévoyez d'ajouter plus d'une connexion à votre Job en cours.
	Close Connection	Décochez cette case pour continuer à utiliser la connexion une fois que le composant a exécuté sa tâche.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
SQL Template	*SQLTemplate List*	Cliquez sur le bouton **Add** pour ajouter un ou plusieurs modèle SQL du système à la liste **SQLTemplateList**. Cliquez dans la cellule de la colonne **SQLTemplate List** que vous venez d'ajouter et sélectionnez le modèle SQL du système adéquat dans la liste de modèles fournis par **Talend**. **Note :** Vous pouvez créer vos propres modèles SQL et les ajouter à la liste **SQLTemplate List**. Pour créer un nouveau modèle SQL : 1- Sélectionnez un modèle déjà existant dans la liste **SQLTemplate list** et cliquez dans son code dans le champ de code. Une boîte de dialogue s'ouvrira vous demandant de créer un nouveau modèle. 2- Cliquez sur **Yes** pour ouvrir la boîte de dialogue **New SQLTemplate**. 3- Renseignez les informations de votre modèle SQL et cliquez sur **Finish** pour fermer la boîte de dialogue. Un éditeur de modèle SQL apparaît dans lequel vous pouvez saisir votre code. 4- Cliquez sur le bouton **Add** pour ajouter le nouveau modèle dans la liste **SQLTemplate list**. Pour plus d'informations, consultez la section *Utilisation des SQL Templates, page 159* du **Guide Utilisateur Talend Open Studio**.
Utilisation	Il faut utiliser ce composant en association avec des composants ELT, notamment avec le **tELTCommit** et le composant ELT de connexion correspondant à la base de données avec laquelle vous travaillez.	

Limitation	n/a

Scénario associé

Pour un scénario associé au **tELTRollback**, consultez le *Scénario : Annuler l'insertion de données dans des tables mère/fille, page 522* du composant **tMySQLRollback**.

tELTSybaseInput

Propriétés de tELTSybaseInput

Les composants **tELTSybaseInput**, **tELTSybaseOutput**, et **tELTSybaseMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base Sybase afin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille de composant	ELT	
Fonction	Fournit le schéma de la table d'entrée à la commande SQL qui sera exécutée.	
Objectif	Permet d'ajouter autant de tables que nécessaires dans une commande Insert qui peut être complexe.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités. Le schéma est soit local (built-in) soit distant dans le Repository. Le schéma défini est ensuite passé au composant ELT Mapper afin de l'utiliser dans la commande SQL Insert Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Default Table Name	Saisissez le nom de la table par défaut.
	Default Schema Name	Saisissez le nom du schéma Sybase par défaut.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	L'utilisation du composant **tELTSybaseInput** est étroitement liée à celle du **tELTSybaseMap**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénarios associés

Pour des scénarios utilisant le composant **tELTSybaseInput**, consultez les scénarios suivants du composant **tELTMysqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTSybaseMap

Propriétés de tELTSybaseMap

Les composants **tELTSybaseInput**, **tELTSybaseOutput**, et **tELTSybaseMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base Sybase afin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille de composant	ELT	
Fonction	Permet de construire votre commande SQL graphiquement en utilisant la table fournie en entrée.	
Objectif	Utilise les tables fournies en entrée pour alimenter les paramètres nécessaires à l'exécution de la commande SQL souhaitée. La commande peut contenir des jointures internes comme externes entre différentes tables ou entre une table et ses alias.	
Basic settings	*Use an existing connection*	Cochez cette case et sélectionnez le composant **tSybaseConnection** adéquat à partir de la liste **Component** pour réutiliser les paramètres d'une connexion que vous avez déjà définie. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	ELT Sybase Map Editor	L'éditeur ELT du Mapper vous permet de définir le schéma de sortie ainsi que de construire graphiquement la commande SQL à exécuter.

	Style link	Sélectionnez le type d'affichage des liens. **Auto** : par défaut, les liens entre les schémas d'entrée et de sortie et les paramètres du service Web sont en forme de courbe. **Curves** : les liens entre les schémas et les paramètres du service Web sont en forme de ligne. **Line** : les liens entre les schémas et les paramètres du service Web sont en forme de lignes droites. Cette option vous permettra d'optimiser les performances.
	Property type	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données
	Port	Numéro de port d'écoute du serveur de la base.
	Database	Nom de la base de données
	Username et *Password*	Données d'authentification de l'utilisateur de la base
	Encoding Type	Sélectionnez l'encodage dans la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour le traitement des données d'une base de données.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	L'utilisation du composant **tELTSybaseMap** est étroitement liée à celle des composants **tELTSybaseInput** et **tELTSybaseOutput**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénarios associés

Pour des scénarios associés, consultez les scénarios suivants du composant **tELTMysqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTSybaseOutput

Propriétés de tELTSybaseOutput

Les composants **tELTSybaseInput**, **tELTSybaseOutput**, et **tELTSybaseMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base Sybase fin de générer des commandes Insert, y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille du composant	ELT	
Fonction	Effectue l'action sur la table telle que spécifiée et insère les données selon le schéma de sortie défini dans le Mapper ELT.	
Objectif	Exécute la requête Insert dans la base de données Sybase.	
Basic settings	*Action on table*	*En Java, utilisez le composant tCreateTable pour remplacer cette option.*
	Action on data	Sur les données de la table définie, vous pouvez effectuer les opérations suivantes : **Insert** : Ajoute de nouvelles lignes à la table. Si des doublons sont identifiés, le Job s'arrête. Note : dans les composants ELT Sybase, seule l'opération Insert est disponible.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Default Table Name	Saisissez le nom de la table par défaut.
	Default Schema Name	Saisissez le nom du schéma Sybase par défaut.
	Use different table name	Cochez cette case afin de définir le nom d'une table de sortie différente, entre guillemets doubles, dans le champ **Table name** qui apparaît.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	L'utilisation du composant **tELTSybaseOutput** est étroitement liée à celle des composants **tELTSybaseInput** et **tELTSybaseMap**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.

Scénarios associés

Pour des scénarios associés, consultez les scénarios suivants du composant **tELTMysqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTTeradataInput

Propriétés de tELTTeradataInput

Les composants **tELTTeradataInput**, **tELTTeradataOutput**, et **tELTTeradataMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base Teradata afin de générer des commandes Insert y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille de composant	ELT	
Fonction	Fournit le schéma de la table d'entrée à la commande SQL qui sera exécutée.	
Objectif	Permet d'ajouter autant de tables que nécessaires dans une commande Insert qui peut être complexe.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités. Le schéma est soit local (built-in) soit distant dans le Repository. Le schéma défini est ensuite passé au composant ELT Mapper afin de l'utiliser dans la commande SQL Insert Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Default Table Name	Saisissez le nom par défaut de la table, entre guillemets doubles.
Utilisation	L'utilisation du composant **tELTTeradataInput** est étroitement liée à celle du **tELTTeradataMap**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénarios associés

Pour des scénarios utilisant le composant **tELTTeradataInput**, consultez les scénarios suivants du **tELTTeradataMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTTeradataMap

Propriétés de tELTTeradataMap

Les composants **tELTTeradataInput**, **tELTTeradataOutput**, et **tELTTeradataMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base Teradata afin de générer des commandes Insert y compris des clauses, et ces commandes devront être exécutées dans la table de sortie définie.

Famille de composant	ELT	
Fonction	Permet de construire votre commande SQL graphiquement en utilisant la table fournie en entrée.	
Objectif	Utilise les tables fournies en entrée pour alimenter les paramètres nécessaires à l'exécution de la commande SQL souhaitée. La commande peut contenir des jointures internes comme externes entre différentes tables ou entre une table et ses alias.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur de base de données
	Port	Numéro de port d'écoute du serveur de la base.
	Database	Nom de la base de données
	Username et *Password*	Données d'authentification de l'utilisateur de la base
	Encoding	Sélectionnez l'encodage dans la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour le traitement des données d'une base de données.
	Preview	L'aperçu est un cliché des données traitées dans le Mapper. Il apparaît lorsque les propriétés du Mapper ont été complétées. La synchronisation de l'aperçu n'est disponible qu'après avoir enregistrer les modifications.
	Map editor	L'éditeur ELT du Mapper vous permet de définir le schéma de sortie ainsi que de construire graphiquement la commande SQL à exécuter.
Utilisation	L'utilisation du composant **tELTTeradataMap** est étroitement liée à celle des composants tELTTeradataInput et **tELTTeradataOutput**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Connecter les composants ELT

Pour plus d'informations concernant la connexion des composants ELT, consultez *Connecter les composants ELT, page 839.*

Voir également : *Connexion de type Link* du **Guide Utilisateur** **Talend Open Studio**

Tables de jointure et de correspondance

Dans le Mapper ELT, vous pouvez sélectionner des colonnes des schémas d'entrée et les intégrer dans le schéma de sortie.

Pour plus d'informations concernant la mise en correspondance et les jointures, consultez *Tables de jointure et de correspondance, page 839.*

Ajouter des clauses Where

Pour plus d'informations sur la manipulation des clauses, consultez *Ajouter des clauses Where, page 839.*

Générer la requête SQL

La mise en correspondance d'éléments des schémas d'entrée avec le schéma de sortie crée automatiquement la requête Select correspondante.

La/les clause(s) définie(s) dans le mapper ELT est (sont) également intégrée(s) automatiquement dans la requête.

Scénarios associés

Pour des scénarios associés, consultez les scénarios suivants du composant **tELTMysqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

tELTTeradataOutput

Propriétés de tELTTeradataOutput

Les composants **tELTTeradataInput**, **tELTTeradataOutput**, et **tELTTeradataMap** sont très proches dans leurs conditions de fonctionnement. En effet, ces trois composants sont conçus pour manipuler des schémas de données d'une base Teradata afin de générer des commandes Insert y compris des clauses, qui devront être exécutées dans la table de sortie définie.

Famille du composant	ELT	
Fonction	Effectue l'action sur la table telle que spécifiée et insère les données selon le schéma de sortie défini dans le Mapper ELT.	
Objectif	Exécute la requête Insert ou Update dans la base de données Teradata	
Basic settings	*Action on data*	Sur les données de la table définie, vous pouvez effectuer les opérations suivantes : **Insert** : Ajoute de nouvelles lignes à la table. Si des doublons sont identifiés, le Job s'arrête. Dans les composants ELT Teradata, seule l'opération Insert est disponible.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Default Table Name	Saisissez le nom par défaut de la table, entre guillemets doubles.
	Use different table name	Cochez cette case afin de définir le nom d'une table de sortie différente, entre guillemets doubles, dans le champ **Table name** qui apparaît.
Utilisation	L'utilisation du composant **tELTTeradataOutput** est étroitement liée à celle des composants **tELTTeradataInput** et **tELTTeradataMap**. Notez que le lien de sortie (output) à utiliser avec ces deux composants doit respecter strictement la syntaxe du nom de la table. Les composants ELT ne traitent pas le flux de données lui-même mais uniquement les informations du schéma.	

Scénarios associés

Pour des scénarios associés, consultez les scénarios suivants du composant **tELTMysqlMap** :

- *Scénario 1 : Agréger les colonnes d'une table et appliquer un filtre, page 840.*
- *Scénario 2 : ELT utilisant une table Alias, page 843.*

Composants File

Ce chapitre passe en revue les propriétés des principaux composants présents dans la famille **File** de la **Palette** de **Talend Open Studio**.

La famille **File** regroupe les composants qui lisent et écrivent des données dans tous types de fichiers à un seul ou plusieurs schémas, du plus courant au format le plus spécifique (dans les sous-familles **Input** et **Output**).

De plus, la sous-famille **Management** regroupe des composants dédiés aux fichiers qui peuvent effectuer diverses traitements sur les fichiers, notamment la décompression, la suppression, la copie, la comparaison de fichiers, etc.

tAdvancedFileOutputXML

tAdvancedFileOutputXML appartient à deux familles de composants : File et XML. Pour plus d'informations concernant le composant **tAdvancedFileOutputXML**, consultez *tAdvancedFileOutputXML, page 1440*.

tApacheLogInput

Propriétés du tApacheLogInput

Component family	File/Input	
Fonction	Le composant **tApacheLogInput** permet de lire le fichier de log d'accès d'un serveur Apache.	
Objectif	Le **tApacheLogInput** aide à la gestion effective du serveur Apache. Il est obligatoire lorsqu'il s'agit d'obtenir un retour sur l'activité et les performances du serveur, mais aussi sur les problèmes susceptibles de se produire.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	File Name	Chemin d'accès et nom du fichier, et/ou variable à traiter. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
Advanced settings	*Encoding*	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du Job, ainsi qu'au niveau de chaque composant.

Utilisation	Ce composant peut être utilisé en standalone. Il permet de créer un flux de données à l'aide d'un lien **Row** > **Main**, ainsi que de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données dont le type ne correspond pas au type défini. Pour un exemple d'utilisation de ces deux liens, consultez le *Scénario 2 : Extraire les données XML erronées dans un flux de rejet, page 1466* du composant **tFileInputXML**.
Limitation	n/a

Scénario : Lire un fichier de log d'accès Apache

Le scénario suivant crée un Job à deux composants permettant de lire le fichier de log d'accès d'un serveur Apache et d'afficher les données en sortie dans la console de log **Run**.

- A partir de la **Palette**, cliquez-déposez les composants **tApacheLogInput** et **tLogRow** dans l'éditeur graphique.
- Cliquez droit sur le **tApacheLogInput** et sélectionnez une connexion **Row** > **Main** dans le menu contextuel. Cliquez ensuite sur le **tLogRow**, le lien approprié apparaît entre les deux composants.

- Dans l'éditeur graphique, sélectionnez le **tApacheLogInput**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du composant **tApacheLogInput**.

- Configurer les champs **Property Type** et **Schema** en mode **Built-In**.
- Cliquez sur le bouton **Edit schema** si vous souhaitez voir les colonnes (en mode read-only).
- Dans le champ **File Name**, renseignez le chemin d'accès ou parcourez vos dossiers jusqu'au fichier de log d'accès que vous voulez lire.

- Dans l'éditeur graphique, sélectionnez le composant **tLogRow** et cliquez sur la vue **Component** pour un définir la configuration de base (**Basic settings**). Pour plus d'informations, consultez le composant *tLogRow, page 1163*.
- Appuyez sur **F6** pour exécuter le Job.

```
Starting job apach at 12:11 01/09/2008.
194.84.172.4|-|-|27/May/2007|03:06:12|+0200|GET|/forum/index.php|HTTP/
1.0|200|19024|http://www.talendforge.org/forum/index.php|Mozilla/4.0
(compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.1.4322)
88.154.28.65|-|-|27/May/2007|05:44:21|+0200|GET|/bugs/view.php?id=205|
HTTP/1.0|200|19456|http://le-gall.net/pierrick/blog/|Mozilla/4.0
(compatible; MSIE 5.0; Windows 98)
72.9.105.42|-|-|27/May/2007|08:53:01|+0200|GET|/forum/register.php|HT
TP/1.0|200|21669|http://www.talendforge.org/register.php|Mozilla/4.0
(compatible; MSIE 7.0b; Windows NT 6.0)
76.171.68.99|-|-|27/May/2007|19:12:33|+0200|GET|/forum/login.php|HTTP/
1.0|200|17445|http://talendforge.org/login.php|Mozilla/4.0
(compatible; MSIE 6.0; Windows NT 5.1) Opera 7.54 [en]
Job apach ended at 12:11 01/09/2008. [exit code=0]
```

Les lignes de log du fichier défini sont affichées dans la console.

tCreateTemporaryFile

Propriétés du tCreateTemporaryFile

Famille du composant	File/Management	
Fonction	Le composant **tCreateTemporaryFile** permet de créer et d'organiser les fichiers temporaires.	
Objectif	Le **tCreateTemporaryFile** aide à la création d'un fichier temporaire et à sa redirection dans dossier défini. Ce composant permet également de conserver ou de supprimer ce fichier temporaire, selon le cas, après l'exécution du Job.	
Basic settings	*Remove file when execution is over*	Cochez cette case pour que le fichier temporaire soit supprimé après l'exécution du Job.
	Directory	Renseignez le chemin d'accès au dossier où sont stockés les fichiers temporaires.
	Template	Attribuez un nom au fichier temporaire en respectant le modèle.
	Suffix	Entrez une extension de fichier pour préciser le format que vous voulez donner au fichier temporaire.
Utilisation	Le composant **tCreateTemporaryFile** vous donne la possibilité de gérer vos fichiers temporaires pour que la mémoire soit libérée au profit d'autres projets, ce qui permet d'optimiser les performances système.	
Global Variables		**Filepath** : retrouve le chemin d'accès à l'emplacement où a été créé le fichier. Disponible comme variable **After**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate. **Trigger** : Run if, OnSubjobOk, OnSubjobError, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur Talend Open Studio**.

Scénario : Créer un fichier temporaire et y écrire des données

Le scénario suivant décrit un Job simple permettant de créer un fichier temporaire vide dans un dossier défini, puis d'y écrire des données et enfin de le supprimer après l'exécution du Job.

- A partir de la **Palette**, cliquez-déposez les composants suivants dans l'éditeur graphique : le **tCreateTemporaryFile**, le **tRowgenerator**, le **tFileOutputDelimited**, le **tFileInputDelimited** et le **tLogRow**.
- Reliez le **tCreateTemporaryFile** au **tRowGenerator** à l'aide d'un lien de type **SubJobOk**.
- Reliez le **tRowGenerator** au **tFileOutputDelimited** à l'aide d'un lien de type **Row Main**.
- Reliez le **tRowGenerator** au **tFileInputDelimited** à l'aide d'un lien de type **SubJobOk**.
- Reliez le **tFileInputDelimited** au **tLogRow** à l'aide d'un lien de type **Row Main**.

- Dans l'éditeur graphique, sélectionnez le composant **tCreateTemporaryFile**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tCreateTemporaryFile**.

- Cochez la case **Remove file when execution is over** pour supprimer, après l'exécution du Job, le fichier temporaire créé.
- Cliquez sur le bouton **[...]** à côté du champ **Directory** pour parcourir vos dossiers jusqu'à celui où seront stockés les fichiers temporaires. Vous pouvez également entrer directement le chemin d'accès.

- Dans le champ **Template**, renseignez le nom du fichier temporaire en veillant à ce qu'il respecte le format du modèle.
- Dans le champ **Suffix**, renseignez l'extension du fichier pour indiquer le format que vous voulez donner au fichier temporaire.
- Dans l'éditeur graphique, sélectionnez le **tRowGenerator** puis cliquez sur la vue **Component** pour en définir la configuration de base (**Basic settings**).

- Dans la liste déroulante **Schema**, choisissez le mode **Built-In**.
- Cliquez sur le bouton **[...]** du champ **Edit schema** pour définir les données à passer au composant **tFileOutputDelimited**. Dans cet exemple, seules les données de la colonne *value* sont traitées.

- Cliquez sur **OK** pour fermer la boîte de dialogue.
- Cliquez sur le bouton **[...]** du champ **RowGenerator Editor** pour ouvrir la boîte de dialogue de l'éditeur.

- Dans le champ **Number of Rows for Rowgenerator**, entrez le chiffre "5" pour générer cinq lignes puis cliquez sur **Ok** pour fermer la boîte de dialogue.

- Dans l'éditeur graphique, sélectionnez le **tFileOutputDelimited** puis cliquez sur la vue **Component** pour en définir la configuration de base (**Basic settings**).

- Dans la liste déroulante **Property Type**, choisissez le mode **Built-In**.
- Cliquez dans le champ **File Name** puis appuyez simultanément sur **Ctrl+Espace** pour accéder à la liste d'auto-complétion. Pour envoyer les données en sortie vers le fichier temporaire créé, sélectionnez l'option `tCreateTemporaryFile_1.FILEPATH` dans la liste des variables.
- Renseignez les séparateurs de lignes et de champs dans leurs champs respectifs **Row Separator** et **Field Separator**.
- Choisissez le mode **Built-In** dans la liste déroulante **Schema** puis cliquez sur **Sync columns** pour synchroniser les colonnes d'entrée et de sortie. Notez que la connexion **Row** permet d'alimenter automatiquement le schéma de sortie.

Pour plus d'informations sur les types de schémas, consultez les chapitres *Paramétrer un schéma built-in* et *Paramétrer un schéma du Repository* du Guide Utilisateur de **Talend Open Studio**.

- Dans l'éditeur graphique, sélectionnez le composant **tFileInputDelimited**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tFileInputDelimited**.

- Choisissez le mode **Built-in** dans la liste déroulante **Property Type**.
- Cliquez dans le champ **File Name** puis appuyez simultanément sur **Ctrl+Espace** pour accéder à la liste d'auto-complétion. Pour lire les données du fichier temporaire créé, sélectionnez `tCreateTemporaryFile_1.FILEPATH` dans la liste des variables.

- Renseignez les séparateurs de lignes et de champs dans leurs champs respectifs **Row Separator** et **Field Separator**.
- Dans la liste déroulante **Schema**, choisissez le mode **Built-In** puis cliquez sur le bouton **[...]** du champ **Edit schema** pour définir les données à passer au composant **tLogRow**. Dans cet exemple, le schéma est composé de la seule colonne *value*.
- Enregistrez le Job puis appuyez sur **F6** pour l'exécuter.

```
Starting job tCreateTemporaryFile at 15:46 19/04/2010.

1
2
3
4
5
Job tCreateTemporaryFile ended at 15:46 19/04/2010. [exit code=0]
```

Le fichier temporaire est créé, lors de l'exécution du Job, dans le dossier qui a été défini et les cinq lignes générées sont inscrites dans ce fichier. Le fichier temporaire est supprimé après l'exécution du Job.

tFileArchive

Propriétés du tFileArchive

Famille de composant	File/Management	
Fonction	Le composant **tFileArchive** zippe un ou plusieurs fichier(s) en fonction des paramètres définis et place l'archive créée dans le répertoire souhaité.	
Objectif	Ce composant zippe un ou plusieurs fichier(s) pour un éventuel traitement.	
Basic settings	*Directory*	Répertoire dans lequel le fichier zippé sera placé. **Subdirectories** (en Perl, ce champ se nomme **Include subdirectories**) : Cochez cette case si le ou les répertoire(s) sélectionné(s) contiennent des sous-dossiers.
	Archive file	Répertoire de destination et nom du fichier archive.
	Compress level	Choisissez le niveau de compression que vous souhaitez appliquer. **Best** : la qualité de compression sera optimale mais le temps de compression sera long. **Normal** : qualité et temps de compression intermédiaires. **Fast** : la compression sera rapide mais la qualité de compression sera amoindrie.
	All files	Cochez cette case si vous souhaitez que tous les fichiers du dossier soient zippés. Décochez-la pour spécifier dans le tableau **Files** le ou les fichier(s) que vous désirez zipper. **Filemask** : entrez un nom de fichier ou masque de fichier utilisant un caractère spécial ou une expression régulière.
Java uniquement	*Create directory if not exists*	Cochez cette case pour créer un nouveau dossier si votre dossier de destination n'existe pas.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
Java uniquement	*Overwrite Existing Archive*	Cette case est cochée par défaut. Elle vous permet de sauvegarder une archive en remplaçant l'archive existante. Cependant, si vous décochez la case, une erreur sera rapportée, le remplacement d'archive échouera et la nouvelle archive ne pourra être sauvegardée. Lorsque le remplacement échoue, le Job poursuit son exécution.

Java uniquement	*Encrypt files*	Cochez cette case si vous souhaitez que votre archive soit protégée par un mot de passe. Le champ **Enter Password** apparaît et vous permet de saisir votre mot de passe.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant doit être utilisé en standalone.	
Global Variables		**Archive File Path** : retrouve le chemin d'accès au fichier archive. Disponible comme variable **After**. Retourne une chaîne de caractères. **Archive File Name** : retrouve le nom du fichier archive. Disponible comme variable **After**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Reject, Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject, Iterate. **Trigger** : Run if, OnSubjobOk, OnSubjobError, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	n/a	

Scénario : Zipper des fichiers à l'aide d'un tFileArchive

Ce scénario met en scène un Job formé d'un composant unique. Il a pour but de zipper des fichiers et de les enregistrer dans le répertoire choisi.

- Cliquez-déposez le composant **tFileArchive** de la **Palette** dans l'espace de modélisation graphique.
- Double-cliquez sur le composant afin d'afficher sa vue **Component**.

- Dans le champ **Directory**, cliquez sur **[...]**, parcourez votre répertoire et sélectionnez le dossier ou le fichier que vous souhaitez compresser.
- Cochez la case **Subdirectories** si vous désirez inclure dans l'archive les fichiers des sous-dossiers.
- Renseignez ensuite le champ **Archive file**, en indiquant le répertoire de destination et le nom de votre archive.
- Cochez la case **Create directory if not exists** si votre répertoire de destination n'existe pas encore et que vous souhaitez le créer.
- Dans la liste **Compress level**, sélectionnez le niveau de compression que vous désirez appliquer à votre archive. Dans cet exemple, utilisez le niveau **normal.**
- Décochez la case **All Files** si vous ne souhaitez compresser que des fichiers spécifiques.

- Ajoutez une ligne dans le tableau à l'aide du bouton **[+]**, et cliquez sur le nom qui s'affiche. Saisissez entre étoiles une partie du nom du ou des fichier(s) que vous souhaitez compresser.
- Appuyez sur **F6** afin d'exécuter votre Job.

Le **tArchiveFile** a compressé les fichiers sélectionnés et a créé le dossier dans le répertoire indiqué.

tFileCompare

Propriétés du tFileCompare

Famille de composant	File/Management	
Fonction	Le composant tFileCompare compare deux fichiers et fournit des informations de comparaison (en fonction d'un schéma en lecture seule).	
Objectif	Ce composant permet de contrôler la qualité des données des fichiers traités.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository mais pour ce composant le schéma est en lecture seule.
	File to compare	Le chemin d'accès au fichier à vérifier.
	Reference file	Le chemin d'accès du fichier sur lequel la comparaison est basé.
	If differences are detected, display *If no difference detected, display*	Saisissez le message à afficher dans la console Run en fonction du résultat de la comparaison.
	Print to console	Cochez cette case pour afficher le message saisi.
Utilisation	Ce composant peut être utilisé en standalone mais il est généralement relié à un composant de sortie pour regrouper les informations de comparaison.	
Global Variables		**Difference** : Vérifie si deux fichiers sont identiques ou non. Disponible comme variable **Flow**. Retourne une valeur booléenne : -true si les deux fichiers sont identiques. -false s'il y a une (des) différence(s) entre eux. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur Talend Open Studio**.

Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject, Iterate. **Trigger** : Run if, OnSubjobOk, OnSubjobError, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	n/a	

Scénario : Comparer des fichiers dézippés

Ce scénario décrit un Job permettant de dézipper un fichier et de le comparer avec un fichier de référence pour s'assurer qu'il n'a pas été modifié. La sortie de cette comparaison est stockée dans un fichier délimité et un message est affiché dans la console.

- Cliquez et déposez les composants suivants : **tFileUnarchive**, **tFileCompare** et **tFileOutputDelimited**.
- Reliez les composants **tFileUnarchive** et **tFileCompare** à l'aide d'une connexion de type **Iterate**.
- Connectez le composant **tFileCompare** au composant de sortie à l'aide d'un lien **Main** row.
- Dans l'onglet **Basic settings** du composant **tFileUnarchive**, renseignez le chemin d'accès au fichier à dézipper.
- Dans le champ **Extraction Directory**, renseignez le chemin de destination du fichier dézippé.
- Dans l'onglet Basic settings du composant **tFileCompare**, paramétrez les propriétés du fichier à comparer. Dans le champ **File to compare**, appuyez sur **Ctrl+Espace** pour afficher la liste des variables globales. Sélectionnez *$_globals{tFileUnarchive_1}{CURRENT_FILEPATH}* ou *"((String)globalMap.get("tFileUnarchive_1_CURRENT_FILEPATH"))"* en fonction du langage avec lequel vous travaillez, pour récupérer le chemin d'accès du fichier à partir du composant **tFileUnarchive**.

- Puis dans le champ **Reference file**, paramétrez le fichier de référence sur lequel est basé la comparaison.
- Dans les champs **If differences detected, display** et **If no differences detected, dispay**, paramétrez les messages à afficher lorsque les fichiers sont différents et lorsqu'ils sont identiques, par exemple : `'[Job '.$_globals{Job_name}.'] Files differ'` si vous travaillez en Perl ou `"[Job " + JobName + "] Files differ"` si vous travaillez en Java.
- Cohez la case **Print to Console**, pour que le message défini apparaisse à la fin de l'exécution du Job.
- Le schéma est en lecture seule et contient les informations standard. Cliquez sur **Edit schema** pour le consulter.

Column	Key	Type	Nullable	Length	Precision	Comment
file		String	✓	255		
file_ref		String	✓	255		
moment		Day	✓			
job		String	✓	50		
component		String	✓	255		
differ		int	✓	1		
message		String	✓	255		

- Puis paramétrez le composant de sortie en utilisant des points-virgules comme séparateurs de données.
- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

```
file;file_ref;moment;job;component;differ;message
C:\Input\Accounts\Sunnyvale_accounts_new.xls;C:/Input/Sunnyv
2007-06-19 14:11:59;CompareFiles;tFileCompare_1;1;[job Compa
differ
```

Le message paramétré apparaît dans la console et la sortie contient les informations telles qu'elles ont été définies dans le schéma.

```
file;file_ref;moment;job;component;differ;message
C:\Input\Accounts\Sunnyvale_accounts_new.xls;C:/Input/Sunnyvale_accounts.csv;
2007-06-19 14:11:59;CompareFiles;tFileCompare_1;1;[job CompareFiles] Files
differ
```

tFileCopy

Propriétés du tFileCopy

Famille de composant	File/Management	
Fonction	Le composant **tFileCopy** fait une copie d'un fichier source dans un répertoire cible et peut supprimer le fichier source, si nécessaire.	
Objectif	Ce composant permet de rationnaliser les traitements en effectuant des tâches récurrentes et fastidieuses de manière automatique.	
Basic settings	*File Name*	Chemin d'accès au fichier à copier ou déplacer.
	Destination	Chemin d'accès du répertoire dans lequel le fichier est copié ou déplacé.
	Remove source file	Cochez cette case pour déplacer le fichier.
	Replace existing file	Cochez cette case pour écraser le fichier déjà existant.
Utilisation	Ce composant peut être utilisé en standalone.	
Global Variables		**Destination File Name** : retrouve le nom du fichier de destination. Disponible comme variable **After**. Retourne une chaîne de caractères. **Destination File Path** : retrouve le chemin d'accès au fichier de destination. Disponible comme variable **After**. Retourne une chaîne de caractères. **Source Directory** : retrouve le chemin d'accès au fichier source. Disponible comme variable **After**. Retourne une chaîne de caractères. **Destination Directory** : retrouve le chemin d'accès au répertoire de destination. Disponible comme variable **After**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur Talend Open Studio**.

Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject, Iterate. **Trigger** : Run if, OnSubjobOk, OnSubjobError, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur Talend Open Studio**.
Limitation	n/a	

Scénario : Récupérer un fichier de la corbeille

Ce scénario décrit un Job faisant une boucle sur une liste de fichiers, copie chaque fichier du répertoire source vers le répertoire cible. Puis il supprime les fichiers copiés du répertoire source.

- Cliquez-déposez les composants **tFileList** et **tFileCopy**.
- Reliez-les à l'aide d'une connexion **Iterate**.
- Dans l'onglet **Basic settings** du composant **tFileList**, spécifiez le répertoire sur lequel sera effectuée la boucle.

- Dans le champ **Filemask**, paramétrez le masque du fichier en *"*.txt"* pour ne tenir compte que des fichiers avec cette extension. Dans cet exemple, la casse n'est pas prise en compte, ainsi dans la liste déroulante **Case Sensitive**, sélectionnez l'option **No**.
- Puis sélectionnez le composant **tFileCopy** et paramétrez ses propriétés dans l'onglet **Basic settings**.

- Dans le champ **File Name**, appuyez sur **Ctrl+Espace** pour accéder à la liste des variables.
- Sélectionnez la variable globale `((String)globalMap.get("tFileList_1_CURRENT_FILEPATH"))` si vous travaillez en Java, ou `$_globals{tFileList_1}{CURRENT_FILEPATH}` si vous travaillez en Perl. De cette manière, tous les fichiers du répertoire source peuvent être traités.
- Cochez la case **Remove Source file** pour supprimer le fichier ayant été copié.
- Cochez la case **Replace existing file** pour écraser tout fichier déjà existant dans le répertoire de destination.
- Enregistrez le Job et appuyez sur **F6**.

Les fichiers sont copiés dans le répertoire de destination et supprimés du répertoire source.

tFileDelete

Propriétés du tFileDelete

Famille de composant	File/Management	
Fonction	Le composant **tFileDelete** supprime un fichier d'un répertoire spécifié.	
Objectif	Ce composant permet de rationnaliser les traitements en effectuant des tâches récurrentes et fastidieuses de manière automatique.	
Basic settings	*File Name*	Chemin d'accès au fichier à supprimer.
Utilisation	Ce composant peut être utilisé en standalone.	
Global Variables		**Delete Path** : retrouve le chemin d'accès à l'emplacement où se trouvait le fichier supprimé. Disponible comme variable **After**. Retourne une chaîne de caractères. **Current status** : indique si un élément a été supprimé ou non. Disponible comme variable **Flow**. Retourne une chaîne de caractères et le libellé de la commande de suppression. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject, Iterate. **Trigger** : Run if, OnSubjobOk, OnSubjobError, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur Talend Open Studio**.
Limitation	n/a	

Scénario : Supprimer des fichiers

Ce scénario décrit un Job très simple supprimant des fichiers d'un répertoire donné.

- Cliquez et déposez les composants suivants : **tFileList**, **tFileDelete**, **tJava**.
- Dans le champ **Directory** de l'onglet **Basic settings** du composant **tFileList**, spécifiez le répertoire sur lequel la boucle doit être effectuée.

- Dans le champ **Filemask**, paramétrez le masque de fichier en "**.txt*". Ne cochez aucune case.
- Dans l'onglet **Basic settings** du composant **tFileDelete**, paramétrez le champ **File Name** afin que le fichier sélectionné par le composant **tFileList** soit supprimé. Ceci permet de supprimer tous les fichiers d'un répertoire spécifié.

- Appuyez sur **Ctrl+Espace** pour accéder à la liste des variables globales. En Java, la variable adéquate permettant de récupérer le fichier en cours est : `((String)globalMap.get("tFileList_1_CURRENT_FILEPATH"))`.
- Puis dans le composant **tJava**, définissez le message à afficher dans la sortie standard (la console **Run**). Dans cet exemple Java, saisissez le script suivant dans le champ **Code** :

```
System.out.println(
((String)globalMap.get("tFileList_1_CURRENT_FILE"))
+ " has been deleted!" );
```

- Puis enregistrez le Job et appuyez sur **F6** pour l'exécuter.

```
Starting job FileDel at 18:29 20/06/2007.
16.txt has been deleted!
15.txt has been deleted!
14.txt has been deleted!
13.txt has been deleted!
12.txt has been deleted!
11.txt has been deleted!
10.txt has been deleted!
09.txt has been deleted!
08.txt has been deleted!
07.txt has been deleted!
06.txt has been deleted!
05.txt has been deleted!
04.txt has been deleted!
03.txt has been deleted!
02.txt has been deleted!
01.txt has been deleted!
Job FileDel ended at 18:29 20/06/2007. [exit code=0]
```

Le message défini dans le composant **tJava** apparaît dans la console pour chaque fichier supprimé à l'aide du composant **tFileDelete**.

tFileExist

Propriétés du tFileExist

Famille du composant	File/Management	
Fonction	Le composant **tFileExist** vérifie si un fichier existe ou non.	
Objectif	Le **tFileExists** aide à rationaliser les processus en automatisant les tâches récurrentes et fastidieuses, comme par exemple vérifier si un fichier existe ou non.	
Basic settings	*File Name*	Chemin d'accès au fichier dont vous voulez vérifier l'existence.
Utilisation	Ce composant peut être utilisé en standalone.	
Global Variables		**Exists** : indique si un fichier spécifié existe ou non. Disponible comme variable **Flow**. Retourne une valeur booléenne : -true si le fichier existe. -false si le fichier n'existe pas. **File name** : retrouve le nom et le chemin d'accès d'un fichier. Disponible comme variable **After**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject, Iterate. **Trigger** : Run if, OnSubjobOk, OnSubjobError, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur Talend Open Studio**.
Limitation	n/a	

Scénario : Vérifier l'existence d'un fichier et le créer s'il n'existe pas

Ce scénario décrit un Job simple qui permet de vérifier si le fichier spécifié existe, d'afficher un message à l'écran si le fichier n'existe pas, de lire les données d'un autre fichier d'entrée spécifié et de les écrire dans un fichier délimité en sortie.

- A partir de la **Palette**, cliquez-déposez les composants suivants dans l'éditeur graphique : le **tFileExist**, le **tFileInputDelimited**, le **tFileOutputDelimited** et le **tMsgBox**.
- Reliez d'abord le **tFileExist** au **tFile InputDelimited** à l'aide d'un lien de type **OnSubjobOk**, puis au **tMsgBox** à l'aide d'un lien de type **Run If**.

- Reliez le **tFileInputDelimited** au **tFileOutputDelimite** à l'aide d'un lien de type **Row Main**.
- Dans l'éditeur graphique, sélectionnez le **tFileExist** puis cliquez sur la vue **Component** pour en définir la configuration de base (**Basic settings**).

- Dans le champ **File name**, renseignez le chemin d'accès au fichier ou parcourez vos dossiers à la recherche du fichier dont vous voulez vérifier l'existence.
- Dans l'éditeur graphique, sélectionnez le composant **tFileInputDelimited** puis cliquez sur la vue **Component** pour en définir la configuration de base (**Basic settings**).

- Remplissez le champ **File Name** en parcourant vos dossiers jusqu'au fichier d'entrée que vous voulez lire.
- Définissez les séparateurs de lignes et de champs dans leurs champs respectifs **Row separator** et **Field Separator**.
- Définissez le nombre de lignes d'en-tête (**Header**), de pied de page (**Footer**) et le nombre maximal de lignes à traiter (**Limit**) en fonction de vos besoins. Dans ce scénario, notre tableau comporte une ligne d'en-tête.
- Dans la liste déroulante **Schema**, choisissez le mode **Built-In**. Cliquez ensuite sur le bouton **[...]** du champ **Edit schema** pour définir les données à passer au composant **tFileOutputDelimited**. Définissez les données présentes dans le fichier à lire ("file2" dans ce scénario).

Pour plus d'informations sur les types de schémas, consultez les chapitres *Paramétrer un schéma built-in* et *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.

Le schéma du fichier "file2" est composé de cinq colonnes : *Num*, *Ref*, *Price*, *Quant* et *Tax*.

- Dans l'éditeur graphique, sélectionnez le composant **tFileOutputDelimited**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tFileOutputDelimited**.

- Définissez le champ **Property Type** en mode **Built-in**.
- Dans le champ **File name**, appuyez simultanément sur **Ctrl+Espace** pour accéder à la liste des variables et sélectionner FILENAME.
- Définissez les séparateurs de lignes et de champs dans leurs champs respectifs **Row separator** et **Field Separator**.
- Cochez la case **Include Header** car, dans cet exemple, le fichier “file2” comprend un en-tête.
- Définissez le champ **Schema** en mode **Built-in** puis cliquez sur le bouton **Sync columns** pour synchroniser le schéma du fichier de sortie (file1) avec le schéma du fichier d’entrée (file2).

- Dans l’éditeur graphique, sélectionnez le composant **tMsgBox**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tMsgBox**.

- Cliquez sur le lien **If** pour en afficher les propriétés dans la vue **Basic settings**.
- Dans le panneau **Condition**, appuyez simultanément sur **Ctrl+Espace** pour accéder à la liste de variable et sélectionner EXISTS. Entrez un point d'exclamation avant la variable pour en inverser le sens.

Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

Une boîte de dialogue apparaît pour confirmer que le fichier n'existe pas.

Cliquez sur **OK** pour fermer cette boîte de dialogue et continuer l'exécution du Job. Le fichier manquant, "file1" dans cet exemple, est écrit sous la forme d'un fichier délimité à l'endroit préalablement déterminé.

tFileInputARFF

Propriétés du tFileInputARFF

Famille de composant	File/Input	
Fonction	Le composant **tFileInputARFF** lit un fichier ARFF ligne à ligne et sépare des champs simples.	
Objectif	Ce composant ouvre un fichier et le lit ligne par ligne afin de le diviser en champs et d'envoyer ces champs au composant suivant, comme défini par le schéma, via une connexion Row.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant connexion de base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio**.
	File Name	Chemin d'accès et nom du fichier, et/ou variable à traiter. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.

Advanced settings	*Encoding*	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les informations de log au niveau du composant.
Utilisation	Utilisez ce composant pour lire un fichier et séparer les champs à l'aide du séparateur spécifié.	

Scénario : Afficher le contenu d'un fichier ARFF

Le scénario suivant décrit un Job formé de deux composants ayant pour objectif de lire ligne à ligne un fichier ARFF, de sélectionner des données délimitées et d'afficher la sortie dans la console de la vue **Run**.

Un fichier ARFF se présente comme suit :

```
@relation vote
@attribute 'handicapped-infants' { 'n', 'y'}
@attribute 'water-project-cost-sharing' { 'n', 'y'}
@attribute 'adoption-of-the-budget-resolution' { 'n', 'y'}
@attribute 'physician-fee-freeze' { 'n', 'y'}
@attribute 'el-salvador-aid' { 'n', 'y'}
@attribute 'religious-groups-in-schools' { 'n', 'y'}
@attribute 'anti-satellite-test-ban' { 'n', 'y'}
@attribute 'aid-to-nicaraguan-contras' { 'n', 'y'}
@attribute 'mx-missile' { 'n', 'y'}
@attribute 'immigration' { 'n', 'y'}
@attribute 'synfuels-corporation-cutback' { 'n', 'y'}
@attribute 'education-spending' { 'n', 'y'}
@attribute 'superfund-right-to-sue' { 'n', 'y'}
@attribute 'crime' { 'n', 'y'}
@attribute 'duty-free-exports' { 'n', 'y'}
@attribute 'export-administration-act-south-africa' { 'n', 'y'}
@attribute 'Class' { 'democrat', 'republican'}
@data
'n','y','n','y','y','y','n','n','n','y',?,'y','y','y','n','y','republican'
'n','y','n','y','y','y','n','n','n','n','n','y','y','y','n',?,'republican'
?,'y','y',?,'y','y','n','n','n','n','y','n','y','y','n','n','democrat'
```

Un fichier ARFF est généralement composé de deux parties : la description de la structure des données, correspondant aux lignes qui débutent par `@attribute`, et les données brutes, qui suivent l'expression `@data`.

Dans l'espace de modélisation du studio de **Talend Open Studio**, créez le Job suivant :

- Glissez et déposez le composant **tFileInputARFF** de la **Palette** dans l'espace de modélisation graphique.
- Glissez et déposez de la même manière le composant **tLogRow**
- Cliquez-droit sur le composant **tFileInputARFF** et sélectionnez **Row > Main** dans le menu contextuel.
- Puis glissez ce lien vers le **tLogRow** et relâchez lorsque le symbole de prise de courant apparaît. La transmission du flux de donnée entre les deux composants est ainsi établie.
- Double-cliquez sur le **tFileInputARFF** pour paramétrer ce composant.
- Dans la vue **Component**, renseignez le champ **File Name**. Parcourez votre répertoire de fichiers, afin de sélectionner le fichier *.arff*.
- Dans le champ **Schema**, sélectionnez **Built-In**.
- Cliquez sur le bouton **[...]** du champ **Edit schema**, pour effectuer manuellement la description des colonnes du fichier lu.
- Cliquez autant de fois sur le bouton que nécessaire pour décrire chacune des colonnes du fichier lu. Dans cet exemple, 17 colonnes sont ajoutées, nommez-les comme suit :

- Pour chacune des colonnes sélectionnez le **Type** : chaine de caractères (**String**).
- Cliquez sur **OK**.

- Dans l'espace de modélisation, double-cliquez sur le composant **tLogRow** pour afficher sa vue **Component**. Cliquez sur le bouton **[...]** du champ **Edit schema** pour vérifier que le schéma a bien été propagé. Si ce n'est pas le cas, cliquez sur le bouton **Sync columns**.

- Conservez les paramètres par défaut.
- Appuyez sur **F6** pour exécuter votre Job.

```
n|n|n|y|y|y|y|y|n|y|n|y|y|y|n|y|republican
n|n|n|n|n|n|y|y|y|y|n|n|y|n|y|y|democrat
y|n|y|n|n|n|y|y|y|y|n|y|n|n|y|y|democrat
n|n|y|y|y|y|n|n|y|y|n|y|y|y|n|y|republican
n|n|y|n|n|n|y|y|y|y|n|n|n|n|n|y|democrat
n|n|n|y|y|y|n|n|n|n|y|y|y|y|n|y|republican
n|n|n|y|y|y|n|n|n|n|n|y|y|y|n|y|republican
n|y|n|y|y|y|n|n|n|y|n|y|y|y|n|n|republican
[statistics] disconnected
Job Test ended at 13:36 16/11/2009. [exit code=0]
```

La console affiche les données contenues dans le fichier ARFF délimitées par une barre verticale (séparateur par défaut).

tFileInputDelimited

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tFileInputDelimited

Famille de composant	File/Input	
Fonction	Le composant **tFileInputDelimited** lit un fichier ou un flux de données ligne par ligne et sépare des champs simples.	
Objectif	Ce composant lit un fichier ou un flux de données ligne par ligne, afin de le diviser en champs et d'envoyer ses champs au composant suivant, comme défini par le schéma, via une connexion Row.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File name/Stream	**File name** : Chemin d'accès et nom du fichier, et/ou variable à traiter. **Stream** : Flux de données à traiter. Les données doivent préalablement être mises en flux afin d'être récupérées par le **tFileInputDelimited** via la variable *INPUT_STREAM* disponible dans la liste d'auto-complétion (**Ctrl+Espace**). Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Row separator	Chaîne (ex : "\n"sous Unix) séparant les lignes.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	CSV options	Cochez cette case pour prendre en compte les paramètres spécifiques aux fichiers CSV, notamment la manière de protéger les caractères dans le champ **Escape char** et le type de guillemet dans le champ **Enclosure**.
	Header	Nombre de lignes à ignorer au début d'un fichier. Lorsque vous utilisez la fonction de schéma dynamique, définissez le champ **Header** comme 0 ou 1, pour utiliser la première ligne du fichier en tant qu'en-tête de la table dynamique de sortie. Pour plus d'informations, consultez la section *Onglet Dynamic settings* dans le **Guide Utilisateur Talend Integration Suite Studio**.
	Footer	Nombre de lignes à ignorer à la fin d'un fichier.

	Limit	Nombre maximum de lignes à traiter. Si Limit = 0, aucune ligne n'est lue ni traitée.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent. Ce composant possède l'avantage d'avoir la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues à partir de fichiers sources ou de copier des colonnes par lots à partir d'un fichier source, sans avoir à les mapper individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez la section *Onglet Dynamic settings* dans le **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Skip empty rows	Cochez cette case pour ignorer les lignes vides.
	Uncompress as zip file	Cochez cette case pour décompresser le fichier d'entrée.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
Advanced settings	*Advanced separator (for numbers)*	Cochez cette option pour modifier les séparateurs utilisés pour les nombres : **Thousands separator** : définissez le séparateur utilisé pour les milliers. **Decimal separator** : définissez le séparateur utilisé pour les décimaux.
	Extract lines at random	Cochez cette case pour définir le nombre de lignes à extraire de façon aléatoire.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	Trim all columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes.

	Check each row structure against schema	Cochez cette case pour synchroniser chaque ligne avec le schéma d'entrée.
	Check columns to trim	Cochez la case devant le nom de chacune des colonnes dont vous voulez supprimer les espaces de début et de fin de champ.
	Split row before field	Cochez cette case pour séparer les lignes avant de séparer les champs.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Utilisez ce composant pour lire un fichier et séparer les champs à l'aide du séparateur spécifié. Il permet de créer un flux de données à l'aide d'un lien **Row** > **Main**, ainsi que de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données dont le type ne correspond pas au type défini. Pour un exemple d'utilisation de ces deux liens, consultez le *Scénario 2 : Extraire les données XML erronées dans un flux de rejet, page 1466* du composant **tFileInputXML**.	

Scénario 1 : Afficher le contenu d'un fichier délimité

Le scénario suivant est un Job de deux composants ayant pour objectif de lire les lignes d'un fichier, de sélectionner des données délimitées et d'afficher la sortie dans la console de la vue **Run**.

- Cliquez et déposez un composant **tFileInputDelimited** de la Palette dans l'espace de modélisation.
- De la même manière, cliquez et déposez un composant **tLogRow**.
- Cliquez-droit sur le composant **tFileInputDelimited** et sélectionnez **Row** > **Main** dans le menu contextuel. Puis glissez ce lien vers le **tLogRow** et relâchez lorsque le symbole de prise de courant apparaît.
- Sélectionnez le composant **tFileInputDelimited** à nouveau et définissez ses propriétés dans l'onglet **Basic Settings** :

- Renseignez le chemin d'accès au fichier dans le champ **File Name**. Ce champ est obligatoire.
- Définissez le séparateur de lignes dans le champ **Row separator** afin d'identifier la fin d'une ligne. Puis définissez le séparateur de champs dans **Field separator** pour délimiter les champs composant une ligne.
- Dans ce scénario, l'en-tête (**Header**) et le pied de page (**Footer**) n'ont pas besoin d'être définis. Et la limite de lignes lues (**Limit**) est de 50 pour cet exemple.
- Sélectionnez soit local (**Built-in**) soit distant (**Repository**) comme **Schema** pour définir les données qui passent par le composant **tLogRow**.
- Vous pouvez charger et/ou éditer le schéma à l'aide de la fonction **Edit Schema**.

Voir également : *Paramétrer un schéma built-in* et *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.

- Assurez-vous que la case **Skip empty rows** est cochée afin d'ignorer les lignes vierges.
- Saisissez l'encodage utilisé dans le fichier d'entrée. Ce paramètre permet d'assurer l'homogénéité de l'encodage dans tous les fichiers d'entrée et de sortie.
- Sélectionnez le composant **tLogRow** et définissez le séparateur de champs de la sortie affichée. Voir également : *Propriétés du tLogRow, page 1163*.
- Cochez la case **Print schema column name in front of each value** pour récupérer le libellé des colonnes dans la sortie affichée.
- Cliquez sur la vue **Run**, puis cliquez sur **Run** pour exécuter le Job.

Le fichier est lu ligne par ligne et les champs extraits sont affichés dans la console, tel que défini dans les propriétés du composant (dans l'onglet **Basic settings**).

```
Starting job FileDel at 10:45 12/04/2007.
ID_Owner: 2|Name: lebouh|ID_Insurance: PKI2906
ID_Owner: 3|Name: bouhnan|ID_Insurance: BNU9147
ID_Owner: 4|Name: hirtle|ID_Insurance: TEV8360
ID_Owner: 5|Name: bobouh|ID_Insurance: WPM3151
ID_Owner: 6|Name: carbone|ID_Insurance: IFY9885
```

La console affiche l'en-tête suivi des données lues par le Job.

Scénario 2 : Lire les données d'un fichier distant en mode stream

Le scénario suivant illustre un Job à quatre composants ayant pour objectif de récupérer les données d'un fichier volumineux quasiment aussitôt qu'elles ne sont lues afin d'éviter l'attente du téléchargement de l'ensemble du fichier de données, et les affichent dans la console de la vue **Run**.

- Déposez les composants suivants dans l'espace de modélisation : **tFileFetch**, **tSleep**, **tFileInputDelimited**, et **tLogRow**.
- Reliez les composants **tSleep** et **tFileInputDelimited** via un lien de type **Trigger** > **OnComponentOk** et reliez les composants **tFileInputDelimited** et **tLogRow** via un lien de type **Row** > **Main**.
- Double-cliquez sur le composant **tFileFetch** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés.

- Dans la liste **Protocol**, sélectionnez le type de protocole vous permettant d'accéder au serveur sur lequel est stocké votre fichier de données.
- Dans le champ **URI**, saisissez l'URI d'accès au serveur sur lequel est stocké votre fichier.

- Cochez la case **Use cache to save the resource** pour mettre les données de votre fichier en mémoire cache. Cette option permet d'utiliser le mode stream pour transférer les données du fichier.
- A partir de l'espace de modélisation, double-cliquez sur le composant **tSleep** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés.
- Par défaut, le champ **Pause** du composant **tSleep** correspond à *1* seconde. Laissez ce paramètre par défaut. Il met le deuxième Job en pause afin de laisser le temps au premier Job contenant le **tFileFetch** de lire les données du fichier.
- A partir de l'espace de modélisation, double-cliquez sur le composant **tFileInputDelimited** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés.

- Dans le champ **File name/Stream** :
  - Effacez le contenu par défaut.
  - Faites un **Ctrl+Espace** pour faire apparaître la liste des variables disponibles à partir de ce composant.
  - Sélectionnez la variable *tFileFetch_1_INPUT_STREAM* dans la liste d'auto-complétion, la varaible suivante apparaît dans le champ **Filename** :
  `((java.io.InputStream)globalMap.get("tFileFetch_1_INPUT_STREAM")).`
- Dans la liste **Schema**, sélectionnez **Built-in** et cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour décrire la structure des données du fichier que vous souhaitez récupérer. Le fichier *US_Employees* est composé de six colonnes : *ID*, *Employee*, *Age*, *Address*, *State*, *EntryDate*.

- Cliquez sur le bouton **[+]** pour ajouter les six colonnes et paramétrez-les comme indiqué sur la capture d'écran ci-dessus. Cliquez sur **OK**.
- A partir de l'espace de modélisation, double-cliquez sur le composant **tLogRow** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés.
- Cliquez sur **Edit schema** et assurez-vous que le schéma est récupéré du composant précédent. Sinon, cliquez sur **Sync Columns** pour le récupérer.
- A partir de l'espace de modélisation, cliquez sur la vue **Job** et sur l'onglet **Extra**.

- Cochez la case **Multi thread execution** afin d'exécuter les deux Jobs en parallèle, sachant que le deuxième Job a un retard d'une seconde par rapport au premier comme défini dans le **tSleep**. Cette option permet donc de récupérer les données grâce au **tFileDelimited** quasiment dès leur lecture par le **tFileFetch**.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
105730|Harry QUINCY|32|Woodson Rd., Pierre|FLORIDA|2008-03-09
105731|Ronald EISENHOWER|26|East Calle Primera, Saint
Paul|TEXAS|2008-07-03
105732|Ulysses BUCHANAN|32|North Ventu Park Road,
Lansing|ARKANSAS|2008-02-16
105733|Thomas TAFT|45|Moreno Drive, Boise|WEST VIRGINIA|2008-08-08
105734|John EISENHOWER|45|Steele Lane, Indianapolis|ALASKA|2007-07-17
105735|Andrew QUINCY|33|El Camino Real, Boston|WISCONSIN|2008-08-25
105736|Lyndon ADAMS|49|Harbor Dr, Frankfort|MARYLAND|2008-12-28
105737|George EISENHOWER|38|Monroe Street, Sacramento|UTAH|2008-09-04
105738|Grover EISENHOWER|42|Steele Lane, Pierre|NEVADA|2008-01-17
105739|Chester CARTER|39|Fontaine Road, Augusta|WEST VIRGINIA|2007-12-20
105740|Abraham NIXON|30|Monroe Street, Columbia|NEBRASKA|2008-07-29
Job tFileInputDelimited ended at 17:58 09/02/2010. [exit code=0]
```

Les données sont affichées dans la console dès leur lecture.

Pour un exemple d'utilisation du schéma dynamique dans le **tFileInputDelimited**, consultez :

- *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507*

tFileInputEBCDIC

Ce composant requiert l'utilisation d'un JDK Sun.

Propriétés du tFileInputEBCDIC

Famille de composant	File/Input	
Fonction	Le composant **tFileInputEBCDIC** lit un fichier EBCDIC et extrait les données en fonction du schéma sélectionné.	
Objectif	Ce composant ouvre un fichier et le lit afin d'isoler les données en fonction des schémas décrivant la structure de ce fichier et d'envoyer ses données et schémas au(x) composant(s) suivant(s) via une connexion Row.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Schema(s)	Ajoutez les différents schémas à transmettre en sortie.
	Data file	Sélectionnez le fichier EBCDIC contenant les données à traiter.
	Xc2j file	Sélectionnez le fichier xc2j de transformation.
Utilisation	Utilisez ce composant pour lire un fichier EBCDIC et séparer les données en fonction des schémas identifiés dans ce fichier.	

Scénario : Extraire les données d'un fichier EBCDIC et alimenter une base de données

Ce scénario utilise l'assistant **[Copybook Connection]** guidant l'utilisateur à travers les différentes étapes de création de connexion au fichier Copybook et de récupération des schémas EBCDIC. L'assistant **[Copybook Connection]** est uniquement disponible pour les utilisateurs **Talend Integration Suite**. Si vous êtes utilisateur de **Talend Open Studio** ou **Talend On Demand**, vous devez configurer les propriétés du composant **tFileInputEBCDIC** manuellement.

Le scénario suivant est un Job de quatre composants ayant pour objectif de lire un fichier au format EBCDIC contenant des informations sur les clients et sur leurs transactions financières, d'en extraire et de transformer les données, et de créer deux tables dans une base de données à partir des deux schémas, clients et transactions, extraits du fichier EBCDIC d'origine.

Ce scénario Java utilise l'assistant **EBCDIC Connection** pour créer une connexion au fichier Copybook et générer un fichier xc2j permettant de récupérer et de transformer les différents schémas du fichier.

- Créez une connexion au fichier Copybook décrivant la structure de votre fichier EBCDIC. Dans ce scénario la connexion Copybook est appelée *EBCDIC*.
- Récupérez le schéma du fichier.

Une fois la connexion au fichier Copybook créée et les schémas récupérés à l'aide des assistants EBCDIC et Schema, ils apparaissent sous le noeud **Metadata** > **Copybook**. Ces nouveaux schémas s'appellent *Schema01*, *Schema04* et *Schema05*.

Pour récupérer les différentes structures du fichier et les utiliser dans **Talend Open Studio** :

- A partir du **Repository**, déposez le schéma *Schema01* dans l'éditeur graphique. Cette manipulation créera le composant d'entrée **tFileInputEBCDIC**.
- Déposez le composant **tMysqlOutput** de la **Palette** dans l'espace de modélisation graphique.
- Double-cliquez sur le **tFileInputEBCDIC** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés :

Les champs **Property Type**, **Schema(s)**, **Data file** et **Xc2j file** sont automatiquement renseignés par les informations de la métadonnée. Le champ **Property Type** indique la métadonnée utilisée pour le composant. Le champ **Schema** indique quel schéma est transmis au composant suivant. Le champ **Data file** indique le chemin d'accès au fichier de données EBCDIC. Le champ **Xc2j file** indique le chemin d'accès au fichier permettant d'extraire le schéma décrivant la structure du fichier EBCDIC. Si vous êtes en mode **Built-In**, renseignez ces champs manuellement.

- Dans l'éditeur graphique, cliquez-droit sur le **tFileInputEBCDIC**, sélectionnez **Row** > **row_Schema01_1** dans le menu contextuel puis cliquez sur le **tMysqlOutput** pour relier les composants entre eux.
- Double-cliquez sur le composant **tMysqlOutput** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés :

- Dans la liste **Property Type**, sélectionnez le mode **Repository** et cliquez sur le bouton **[...]** pour sélectionner la connexion à la base de données centralisée dans les métadonnées du Repository. Les champs **Host**, **Port**, **Database**, **Username** et **Password** seront renseignés automatiquement. Si vous êtes en mode **Built-In**, renseignez ces champs manuellement.

- Dans le champ **Table**, renseignez le nom de la table à créer qui contiendra les données extraites du fichier EBCDIC. Ici, appelez cette table *ebcdic_01*.
- Dans le champ **Action on table**, sélectionnez l'option **Create table** pour créer la table.

A ce stade, le Job récupère le schéma *Schema01* du fichier EBCDIC et le transmet ainsi que ses données correspondantes vers la base de données. Vous souhaitez maintenant récupérer le schéma et les données du schéma *Schema04* du fichier EBCDIC, effectuer une transformation sur ces données et les transmettre à cette même base de données. Pour cela :

- Cliquez et déposez les composants **tMap** et **tMysqlOutputBulkExec** dans l'éditeur graphique.
- Double-cliquez sur le composant **tFileInputEBCDIC** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés :

- Dans le champ **Schema(s)**, cliquez sur le bouton **[+]** pour ajouter une ligne.
- Cliquez sur cette ligne puis sur le bouton **[...]** qui apparaît en fin de champ. Dans la boîte de dialogue qui s'ouvre, sélectionnez **Create schema from repository** pour récupérer le schéma défini dans la métadonnée EBCDIC puis dans la liste déroulante, sélectionnnez le schéma *Schema04*.
- Cliquez sur **OK** pour fermer la boîte de dialogue.
- Si vous n'avez pas récupéré le schéma dans les métadonnées du repository, sélectionnez **Create schema for built-in**, puis spécifiez manuellement le nom et la description de votre schéma.
  Les deux schémas *Schema01* et *Schema04* apparaissent dans le champ **Schema(s)** du composant **tFileInputEBCDIC**.
- Pour relier les composants entre eux, cliquez-droit sur le **tFileInputEBCDIC**, sélectionnez **Row** > **row_Schema04_1** dans le menu contextuel puis cliquez sur le **tMap**. Puis cliquez-droit sur le **tMap**, glissez le lien jusqu'au **tMysqlOutputBulkExec** et relachez le bouton. Dans la boîte de dialogue qui s'ouvre, indiquez le nom de la sortie *ebcdic_04*.
- Double-cliquez sur le **tMap** pour ouvrir son éditeur

- Sélectionnez toutes les colonnes de la table **row_Schema04_1** et glissez-les dans la table **ebcdic_04**.
- Dans le tableau **ebcdic_04** de la zone **Schema editor** en bas de l'éditeur, cliquez sur le bouton **[+]** pour ajouter une colonne au schéma. Nommez cette colonne *SUM_AG_NUMBER*.
- Dans la table **row_Schema04_1** à gauche de l'éditeur, appuyez sur **Ctrl** puis sélectionnez les colonnes **CC01404_L_11_MENAG_1_1** et **CC01404_AG_CAM_1_1** et glissez-les dans la nouvelle colonne *SUM_AG_NUMBER* de la table **ebcdic_04**, et ajoutez le signe + entre les deux colonnes concaténées pour obtenir : `row_04_1.CC01404_L_11_MENAG_1_1 + row_04_1.CC01404_AG_CAM_1_1`.
- Cliquez sur **OK** pour validere vos modifications et fermer l'éditeur.
- Dans l'éditeur graphique, double-cliquez sur le **tMysqlOutputBulkExec** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés :

- Dans la liste **Property Type**, sélectionnez le mode **Repository** et cliquez sur le bouton **[...]** pour sélectionner la connexion à la base de données centralisée dans les métadonnées du Repository. Les champs **Host**, **Port**, **Database**, **Username** et **Password** seront renseignés automatiquement. Si vous êtes en mode **Built-In**, renseignez ces champs manuellement.
- Dans le champ **Table**, renseignez le nom de la table à créer qui contiendra les données extraites du fichier EBCDIC.
- Dans le champ **Action on table**, sélectionnez l'option **Create table** pour créer la table.
- Enregistrez votre Job via **Ctrl+S** et cliquez sur la vue **Run**, cochez les cases **Statistics** et **Exec time** puis cliquez sur **Run** pour exécuter le Job.

Les deux tables ont été crées dans la base de données et contiennent la structure ainsi que les données, de transactions et clients, du fichier EBCDIC d'origine.

CC01394_C_ENR	CC01394_C_...	CC01394_IND...	CC01394_D_T...	CC01394_C_4_HH...	CC01394_D_D2_ARRE
01	891	E	20080519	2356	20080430

CC...	CC01404_L...	CC01404_D_...	CC01404_L_11_ID...	CC01404_...	CC014...	CC01404_L_C
04	00000003	20070512	00000035644	19990303	MME	WILSON
04	00000004	20080208	00000035647	19500101	MME	HARRISON
04	00000005		00000035654	19660608	MR	ADAMS
04	00000006	20080326	00000035670	19980804	MME	JACKSON
04	00000007	20080429	00000035671	19660722	MR	JACKSON
04	00000008	20071001	00000035692	19500101	MME	PIERCE
04	00000009	20080123	00000035693	19720826	MR	MCKINLEY

tFileInputExcel

Propriétés du tFileInputExcel

Famille du composant	File/Input	
Fonction	Le composant **tFileInputExcel** lit un fichier Excel (.xls ou .xlsx) ou un flux de données et en extrait les données ligne par ligne.	
Objectif	Ce composant lit un fichier Excel ou un flux de données ligne par ligne pour le scinder en champs et envoie les champs tel que définis dans le schéma au composant suivant du Job via une connexion **Row**.	
Basic settings	*Property type*	Peut être Built-in ou Repository
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de connexion et enregistrer les paramètres de connexion au fichier Excel que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de fichier, consultez le chapitre *Configurer un schéma File Excel* du **Guide Utilisateur** de **Talend Open Studio** .
	File name/Stream	**File name** : Chemin d'accès et nom du fichier, et/ou variable à traiter. **Stream** : Flux de données à traiter. Les données doivent préalablement être mises en flux afin d'être récupérées par le **tFileInputExcel** via la variable *INPUT_STREAM* disponible dans la liste d'auto-complétion (**Ctrl+Espace**). Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	All sheets	Cochez cette case pour traiter toutes les feuilles du fichier Excel.
	Sheet list	Cliquez sur le bouton **[+]** pour ajouter de nouvelles lignes à la liste de feuilles à traiter : **Sheet (name or position)** : renseignez le nom ou la position de la feuille à traiter. **Use Regex** : cochez cette case si vous utilisez une expression régulière pour filtrer les feuilles selon un modèle.
	Header	Nombre de lignes à ignorer au début d'un fichier.

	Footer	Nombre de lignes à ignorer à la fin d'un fichier.
	Limit	Nombre maximum de lignes à traiter. Si Limit = 0, aucune ligne n'est lue ni traitée.
	Affect each sheet(header&footer)	Cochez cette case pour appliquer les paramètres définis dans les champs **Header** et **Footer** à toutes les feuilles traitées.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
	First column et *Last column*	Renseignez l'intervalle de colonnes que vous voulez traiter : la position de la première colonne dans le champ **First column** et celle de la dernière colonne à traiter dans le champ **Last column**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
Advanced settings	*Advanced separator*	Cochez cette case pour modifier les séparateurs de lignes utilisés.
	Trim all columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes. Lorsque cette case est décochée, le tableau **Check column to trim** s'affiche, vous permettant de sélectionner la ou les colonne(s) dont vous souhaitez retirer les espaces de début ou de fin.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
	Read real values for numbers	Cochez cette case pour lire les nombres en valeur réelle.
	Stop to read on empty rows	Cochez cette case pour ignorer les lignes vides.
	Don't validate the cells	Cochez cette case pour ne pas effectuer de validation des données.
	Ignore the warning	Cochez cette case pour ignorer les avertissements générés par des erreurs sur le fichier Excel.

	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Utilisez ce composant pour lire un fichier Excel (.xls ou .xlsx) et séparer les champs en fonction d'un schéma défini. Ce composant permet de créer un flux de données à l'aide d'un lien **Row** > **Main**, ainsi que de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données dont le type ne correspond pas au type défini. Pour un exemple d'utilisation de ces deux liens, consultez le *Scénario 2 : Extraire les données XML erronées dans un flux de rejet, page 1466* du composant **tFileInputXML**.	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tFileInputFullRow

Propriétés du tFileInputFullRow

Famille du composant	File/Input	
Fonction	Le composant **tFileInputFullRow** lit un fichier donné ligne par ligne.	
Objectif	Le **tFileInputFullRow** ouvre un fichier qu'il lit ligne par ligne et passe les lignes complètes au composant suivant comme défini dans le schéma, via une connnexion de type Row.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant **tFileInputFullRow** précédent.
	File Name	Chemin d'accès et nom du fichier, et/ou variable à traiter. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Row separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Header	Nombre de lignes à ignorer en début de fichier.
	Footer	Nombre de lignes à ignorer à la fin d'un fichier.
	Limit	Nombre maximum de lignes à traiter. Si Limit = 0, aucune ligne n'est lue ni traitée.
	Skip empty rows	Cochez cette case pour ignorer les lignes vides.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur.
Utilisation	Utilisez ce composant pour lire des lignes complètes dans un fichier délimité pouvant être très gros. Ce composant permet de créer un flux de données à l'aide d'un lien **Row** > **Main**, ainsi que de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données dont le type ne correspond pas au type défini. Pour un exemple d'utilisation de ces deux liens, consultez le *Scénario 2 : Extraire les données XML erronées dans un flux de rejet, page 1466* du composant **tFileInputXML**.	

Scénario : Lire des lignes complètes dans un fichier délimité

Le scénario suivant crée un Job à deux composants permettant de lire des lignes complètes à partir d'un fichier et d'afficher les données de sortie dans la console de log **Run**.

- A partir de la Palette, cliquez-déposez les composants **tFileInputFullRow** et **tLogRow** dans l'éditeur graphique.
- Cliquez-droit sur le composant **tFileInputFullRow** et connectez-le au **tLogRow** à l'aide d'un lien de type **Row Main**.

- Dans l'éditeur graphique, sélectionnez le composant **tFileInputFullRow**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tFileInputFullRow**.

- Dans l'onglet **Basic settings**, configurez le champ **Schema** en mode **Built-In**.
- Clickez sur le bouton **[...]** du champ **Edit schema** pour voir les données à passer au composant **tLogRow**. Notez que le schéma est en read-only et ne comporte qu'une seule colonne, *line*.

- Dans le champ **File Name**, renseignez le chemin d'accès au fichier à traiter ou cliquez sur le bouton **[...]**. Ce champ est obligatoire. Dans ce scénario, le fichier à lire s'appelle *test5* et comporte trois lignes comprenant chacune deux champs séparés par un point-virgule.
- Dans le champ **Row separator**, renseignez le séparateur permettant d'identifier les fins de ligne.

- Dans la champ **Header**, indiquez la valeur 1. Dans ce scénario, ni le pied-de-page ni le nombre de lignes à traiter ne sont renseignés.
- Dans l'éditeur graphique, sélectionnez le **tLogRow** puis cliquez sur la vue **Component** pour en définir la configuration de base (**Basic settings**). Pour plus d'informations, consultez le *tLogRow, page 1163*.
- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

```
Starting job Input_Full_Row at 11:13 26/08/2008.
.--------------------.
|      tLogRow_1     |
|=------------------=|
|line                |
|=------------------=|
|Janet,Anderson;I9988|
|Martin,Chairman;9889|
|Lily,Massy;9988     |
'--------------------'

Job Input_Full_Row ended at 11:14 26/08/2008. [exit
```

Le composant **tFileInputFullRow** lit les trois lignes une à une sans tenir compte des séparateurs de champs et les lignes complètes apparaissent dans la console **Run**.

Pour extraire des lignes uniquement les champs, utilisez le **tExtractDelimitedFields**, le **tExtractPositionalFields** et le **tExtractRegexFields**. Pour plus d'informations, consultez les composants *tExtractDelimitedFields, page 1276*, *tExtractPositionalFields, page 1281* et *tExtractRegexFields, page 1283*.

tFileInputJSON

Propriétés du tFileInputJSON

Famille de composant	File	
Fonction	Le composant **tFileInputJSON** lit un fichier JSON et extrait les données en fonction du schéma sélectionné.	
Objectif	Ce composant ouvre un fichier et le lit afin d'isoler les données en fonction des schémas décrivant la structure de ce fichier et d'envoyer ses données et schémas au(x) composant(s) suivant(s) via une connexion Row.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement..
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Use URL	Cochez cette case pour récupérer les données directement à partir d'internet. **URL** : entrez le chemin d'accès à l'URL à partir de laquelle récupérer les données.
	Filename	Nom du fichier à partir duquel récupérer les données.
	Mapping	**Column** : reflète le schéma tel qu'il est défini dans l'éditeur Edit Schema. **JSONPath Query** : Saisissez les champs à extraire de la structure JSON d'entrée.
Advanced settings	*Advanced separator*	Cochez cette case pour modifier les séparateurs de lignes utilisés.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.

	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du Job, ainsi qu'au niveau de chaque composant.
Utilisation	Utilisez ce composant pour lire un fichier JSON et séparer les données en fonction des schémas identifiés dans ce fichier.	
Limitation	n/a	

Scénario : Extraire les données des champs d'un fichier au format JSON

Ce scénario est de deux composants et permet de lire un fichier au format JSON et d'en extraire les données.

- Glissez les composants **tFileInputJSON** de la famille **File** et **tLogRow** de la famille **Logs & Errors** à partir de la **Palette** dans le Job designer.
- Reliez les composants entre eux via des connexions de type **Main** > **Row**.
- Double-cliquez sur le composant **tFileInputJSON** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component** :

- Si votre schéma est déjà stocké sous le noeud **Db Connections** dans le **Repository**, sélectionnez l'option **Repository** dans le champ **Schema Type** puis choisissez les métadonnées appropriées à partir de la liste.
- Si vous n'avez encore défini aucun schéma, sélectionnez l'option **Built-in** et renseignez manuellement les informations de connexion et la structure des données dans un schéma.
- Cliquez sur le bouton **[...]** du champ **Edit schema** pour ouvrir une boîte de dialogue dans laquelle vous pouvez définir le schéma à afficher en sortie.

- Cliquez sur le bouton **OK** pour fermer la boîte de dialogue. Dans la zone **Mapping**, les éléments du champ **Column** sont alors renseignés automatiquement selon le schéma que vous venez de définir. Dans cet exemple, le schéma comporte quatre colonnes *FirstName*, *LastName*, *Address* et *City*.
- Dans le champ **Filename**, renseignez le chemin d'accès au fichier JSON dont vous voulez récupérer les données. Si les données sont stockées à distance sur Internet, cochez la case **Use URL** puis, de la même façon, renseignez l'URL d'accès au fichier à traiter. Dans cet exemple, le fichier traité se présente comme suit :

```
{
    "firstName": "John",
    "lastName": "Smith",
    "address": {
        "streetAddress": "21 2nd Street",
        "city": "New York",
        "state": "NY",
        "postalCode": "10021"
    },
    "phoneNumbers": [
        { "type": "home", "number": "212 555-1234"
},
        { "type": "fax", "number": "646 555-4567" }
```

- Dans la zone **Mapping**, les lignes du champ **Column** sont déjà renseignées. Pour chacune d'elle, indiquez dans le champ **JSONPath query** le niveau de l'arborescence sur lequel récupérer les données.
- Dans le Job designer, double-cliquez sur le composant **tLogRow** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component**.

- Cliquez sur le bouton **Sync Columns** pour récupérer le schéma du composant précédent.
- Enregistrez le Job et appuyez sur **F6** pour l'éxecuter.

```
Starting job tFileInputJSON at 13:54 04/11/2009.

[statistics] connecting to socket on port 3663
[statistics] connected
.---------+--------+-------------+--------.
|                tLogRow_1                |
|=--------+--------+-------------+-------=|
|FirstName|LastName|Address      |City    |
|=--------+--------+-------------+-------=|
|John     |Smith   |21 2nd Street|New York|
'---------+--------+-------------+--------'

[statistics] disconnected
Job tFileInputJSON ended at 13:54 04/11/2009. [exit code=0]
```

Le Job retourne les informations du client selon les paramètres sélectionnés dans le schéma.

tFileInputLDIF

Propriétés du tFileInputLDIF

Famille du composant	File/Input	
Fonction	Le composant **tFileInputLDIF** lit un fichier LDIF donné ligne par ligne.	
Objectif	Le **tFileInputLDIF** ouvre un fichier qu'il lit ligne par ligne et passe les lignes complètes au composant suivant comme défini dans le schéma, via une connnexion de type Row.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier où sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Chemin d'accès et nom du fichier, et/ou variable à traiter. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	add operation as prefix when the entry is modify type	Cochez cette case pour afficher le mode d'opérations effectuées.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant **tFileInputLDIF** précédent.
Advanced settings	*Encoding*	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du Job, ainsi qu'au niveau de chaque composant.

Utilisation	Utilisez ce composant pour lire des lignes complètes dans un fichier délimité pouvant être très gros.Ce composant permet de créer un flux de données à l'aide d'un lien **Row** > **Main**, ainsi que de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données dont le type ne correspond pas au type défini. Pour un exemple d'utilisation de ces deux liens, consultez le *Scénario 2 : Extraire les données XML erronées dans un flux de rejet, page 1466* du composant **tFileInputXML**.

Scénario associé

Pour un scénario associé, consultez *Scénario : Ecrire des données dans un fichier LDIF, page 995*.

tFileInputMail

Propriétés du tFileInputMail

Famille de composant	File/Input	
Fonction	Le composant **tFileInputMail** lit l'en-tête et le contenu du fichier e-mail défini.	
Objectif	Ce composant permet d'extraire les données standard des emails.	
Basic settings	*File name*	Entrez le fichier email source.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma dans cette vue, le schéma devient automatiquement ponctuel (built-in). Cliquez sur **Sync columns** pour récupérer le schéma à partir du composant précédent dans le Job.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Attachment export directory	Renseignez le chemin d'accès au répertoire dans lequel exporter les pièces jointes.
	Mail parts	**Column** : Ce champ est automatiquement renseigné avec les colonnes définies dans le schéma que vous avez propagé.
		Mail part : Saisissez l'étiquette d'en-tête ou de corps de message à afficher dans la sortie.
		Multi value *:* Cochez cette case pour chaque colonne comportant des champs à valeurs multiples.
		Field separator *:* Saisissez le séparateur de valeur des champs à valeurs multiples.

	Die on error	Cochez cette case pour arrêter à l'exécution du Job lorsqu'une erreur survient. Décochez la case pour la ligne en erreur et terminer le processus avec les lignes sans erreur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du Job, ainsi qu'au niveau de chaque composant.
Utilisation	Ce composant traite un flux de données et par conséquant requiert un composant de sortie.	
Limitation	n/a	

Scénario : Extraire des champs clés d'un email

Ce scénario est constitué de deux composants et a pour objectif d'extraire certains champs clés et d'en afficher les valeurs dans la console **Run**.

- Cliquez et déposez les composants **tFileInputMail** et **tLogRow**.
- Reliez les composants à l'aide d'un lien de type **Row Main**.
- Double-cliquez sur le composant **tFileInputMail** pour afficher l'onglet **Basic settings** de la vue **Component** et configurer les paramètres de base :

- Dans le champ **File Name**, sélectionnez le fichier email à traiter. Renseignez manuellement le chemin d'accès ou cliquez sur le bouton **[...]** pour parcourir vos dossiers jusqu'au fichier choisi.
- Cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour ouvrir la boîte de dialogue permettant de définir le schéma à afficher en sortie.

- Définissez votre schéma en incluant toutes les colonnes que vous souhaitez retrouver dans le flux de sortie. Dans cet exemple, le schéma comporte quatre colonnes *Date*, *Author*, *Object* et *Status*.
- Une fois défini, cliquez sur **OK** pour fermer la boîte de dialogue et propager le schéma dans le tableau **Mail parts.**
- Dans la colonne **Mail part** du tableau, saisissez les clés standards des parties header (en -tête) et body (corps de message) qui seront utilisées pour retrouver les valeurs à envoyer en sortie.
- Cochez la case **Multi Value** pour chacune des clés standards pouvant se retrouver plus d'une fois dans votre fichier d'entrée. Définissez si besoin un séparateur pour les valeurs de ces clés dans le champ **Separator**.
- Double-cliquez sur le composant **tLogRow** pour afficher l'onglet **Basic settings** de la vue **Component** et configurer les paramètres de base.
- Définissez le composant **tLogRow** afin que les valeurs affichées soient séparées par un retour chariot. Sous Windows, saisissez *\n* entre guillemets doubles. Pour plus d'informations sur le composant **tLogRow**, consultez le chapitre *tLogRow, page 1163*.
- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter et afficher le flux de sortie dans la console de la vue **Run Job**.

```
Starting job tFileInputMail at 11:48 06/11/2009.

[statistics] connecting to socket on port 3437
[statistics] connected
Wed, 4 Nov 2009 19:12:47 +0800
(CST)|musicatcher@gmail.com|Talend multi value test|by
10.142.186.14 with SMTP id j14cs69293wff;
        Wed, 4 Nov 2009 03:13:00 -0800 (PST)by 10.150.45.40
with SMTP id s40mr2413104ybs.260.1257333179981;
        Wed, 04 Nov 2009 03:12:59 -0800 (PST)from
mail-gx0-f210.google.com (mail-gx0-f210.google.com
[209.85.217.210])
        by mx.google.com with ESMTP id
33si1593716ywh.127.2009.11.04.03.12.58;
        Wed, 04 Nov 2009 03:12:58 -0800 (PST)by
mail-gx0-f210.google.com with SMTP id 2so6057662gxk.4
        for <musicatcher0@gmail.com>; Wed, 04 Nov 2009 03:12:58
-0800 (PST)by 10.150.75.12 with SMTP id
x12mr2370266yba.341.1257333175484;
        Wed, 04 Nov 2009 03:12:55 -0800 (PST)from nsun
([219.237.242.224])
        by mx.google.com with ESMTPS id
5sm364705ywd.23.2009.11.04.03.12.53
        (version=SSLv3 cipher=RC4-MD5);
        Wed, 04 Nov 2009 03:12:54 -0800 (PST)
[statistics] disconnected
Job tFileInputMail ended at 11:48 06/11/2009. [exit code=0]
```

Les valeurs clés d'en-tête sont extraites comme spécifié dans le tableau **Mail parts**. En effet, la date de réception, l'auteur, le sujet ainsi que les différents statuts contenus dans l'email sont affichés.

tFileInputMSDelimited

Propriétés du tFileInputMSDelimited

Famille du composant	File/Input	
Fonction	Le composant **tFileInputMSDelimited** permet de lire des fichiers délimités multi structure complexes.	
Objectif	Le **tFileInputMSDelimited** ouvre le fichier multi structure complexe, en lit la structure (schémas) puis utilise des liens de type **Row** pour envoyer les champs, tels qu'ils ont été définis dans les différents schémas, vers les composants suivants dans le Job.	
Basic settings	*Multi Schema Editor*	L'éditeur de schémas **[Multi Schema Editor]** permet de construire et de configurer le flux de données dans un fichier délimité multi structure pour pouvoir associer un schéma pour chaque sortie. Pour plus d'informations, consultez *L'éditeur de schémas Multi Schema Editor, page 943*.
	Output	Les schémas définis dans l'éditeur de schémas **[Multi Schema Editor]**, ainsi que les types d'enregistrements correspondants, s'affichent automatiquement dans le tableau **Output**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur.
Advanced settings	*Trim all columns*	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Utilisez ce composant pour lire des fichiers délimités multi structure et définissez un séparateur de champs pour différencier les champs contenus dans ces fichiers.	

L'éditeur de schémas Multi Schema Editor

L'éditeur **[Multi Shema Editor]** vous permet de définir :

- le chemin d'accès au fichier source,
- les propriétés du fichier source,
- la structure des données pour chacun des schémas de sortie.

Lorsque vous définissez la structure de chaque schéma de sortie dans le **[Multi Schema Editor]**, le nom des colonnes des différentes structures s'affichent automatiquement dans la liste du schéma d'entrée des composants qui suivent le **tFileInputMSDelimited**. Vous pouvez cependant définir la structure des données directement dans la vue **Basic settings** de chacun de ces composants.

Le **[Multi Schema Editor]** permet également de sélectionner un schéma qui servira de source (clé primaire) aux données entrantes, afin d'en assurer l'unicité. L'éditeur utilise ce procédé de

mapping pour associer tous les schémas traités dans le fichier délimité au schéma source de ce même fichier.

L'éditeur s'ouvre en affichant par défaut la première colume. Cette colume contient généralement l'indicateur de type des enregistrements. Une fois ouvert, vous pouvez cependant définir comme clé primaire n'importe quelle colonne du schéma en cochant la case associée.

L'image ci-dessous donne un aperçu de l'éditeur de schémas **[Multi Schema Editor]**.

Pour des informations plus détaillées sur le **Multi Schema Editor**, consultez le *Scénario : Lecture d'un fichier délimité multi structure, page 945*.

Scénario : Lecture d'un fichier délimité multi structure

Le présent scénario sert à créer un Job en Java permettant de lire trois schémas dans un même fichier délimité et d'en afficher la structure sur la console de la vue **Run**.

Le fichier délimité traité dans ce scénario se présente comme suit :

```
01;SOFT MUSIC ALBUM;RICHARDSON;15/12/2005
02;We Danced
02;She's Everything
02;Once in a Lifetime Love
03;National Library
01;COUNTRY MUSIC ALBUM;WHITE;02/01/2006
02;Fall Into Me
02;Another Try
02;Something About Her
```

- A partir de la **Palette,** cliquez sur les composants suivants et déposez-les dans l'éditeur graphique : un composant **tFileInputMSDelimited** et trois **tLogRow**.
- Double-cliquez sur le **tFileInputMSDelimited** pour ouvrir l'éditeur **Multi Schema Editor**.

- Cliquez sur le bouton **Browse...** à côté du champ **File name** pour indiquer où se trouve le fichier délimité multi-schéma que vous voulez traiter.
- Dans la zone **File Settings**, renseignez :
  -le type d'encodage de votre fichier source, à partir de la liste déroulante. Cette option permet de garantir une certaine cohérence entre tous les fichiers d'entrée et de sortie.
  -les séparateurs de champ et de ligne utilisés dans le fichier source.

> Cochez la case **Use Multiple Separator** et configurez les champs qui suivent si différents séparateurs de champs sont utilisés pour séparer les schémas du fichier source.

Un aperçu des données du fichier source apparaît automatiquement dans le panneau d'affichage **Preview**.

La colonne **Column 0**, qui contient généralement l'indicateur de type des enregistrements est sélectionnée par défaut. Vous pouvez cependant définir n'importe quelle autre colonne comme clé primaire en cochant sa case associée.

- Cliquez sur le bouton **Fetch Codes** à droite du panneau **Preview** pour cataloguer les types de schémas et les enregistrements du fichier source. Dans ce scénario, le fichier source comporte trois types de schémas (*A*, *B*, *C*).
- A partir du panneau d'affichage **Fetch Codes**, cliquez sur chacun des trois types pour que la structure de leurs données s'affiche dans le panneau **Preview**.

- Cliquez dans les cellules de la ligne *Name* et choisissez le nom des colonnes de chacun des schémas sélectionnés.
  Dans ce scénario, les colonnes se lisent comme suit :
  -Schéma *A* : *Type*, *DiscName*, *Author*, *Date*,
  -Schéma *B* : *Type*, *SongName*,
  -Schéma *C* : *Type*, *LibraryName*.

Vous devez à présent définir, à partir des données d'entrée, la clé primaire qui permettra de conserver leur unicité (dans ce scénario, il s'agit de *DiscName*). Il vous faut pour cela :

- Sélectionner, à partir du panneau d'affichage **Fetch Codes**, le schéma contenant la colonne que vous voulez définir comme clé primaire (dans ce scénario, il s'agit du schéma *A*) pour en afficher la structure.
- Cliquer dans la cellule *Key* correspondant à la colonne *DiscName* et cocher la case qui s'affiche alors.

- Cliquer n'importe où dans l'éditeur pour que la mention *false*, dans la case *Key*, devienne *true*.

Vous devez maintenant indiquer le schéma "père" auquel vous voulez rattacher les autres schémas "fils" (dans ce scénario, il s'agit de *DiscName*). Il vous faut pour cela :

- Sélectionner le schéma *B*, à partir du panneau d'affichage **Fetch Codes**, puis cliquer sur le bouton flèche droite pour le déplacer vers la droite.
- Faire de même pour le schéma *C*.

Le champ **Cardinality** n'est pas obligatoire. Il vous permet de déterminer le nombre (ou l'intervalle) de champs des schémas "fils" rattachés au schéma "père". Cependant, si vous entrez un nombre ou un intervalle inexact et essayez d'exécuter le Job, un message d'erreur apparaîtra.

- Dans le **[Multi Schema Editor]**, cliquez sur le bouton **OK** pour valider tous les changements que vous venez d'effectuer et fermer l'éditeur de schémas.

Les trois schémas définis, ainsi que les types d'enregistrements correspondants, s'affichent automatiquement dans la vue **Basic settings** du composant **tFileInputMSDelimited**.

- Dans l'éditeur graphique, effectuez un clic-droit sur le composant **tFileInputMSDelimited** et connectez-le aux trois autres composants **tLogRow1**, **tLogRow2** et **tLogRow3** à l'aide des liens respectifs **row_A_1**, **row_B_1** et **row_C_1**.

Les trois schémas que vous avez définis dans l'éditeur **[Multi Schema Editor]** sont automatiquement communiqués aux trois composants **tLogRow**.

- Si nécessaire, cliquez sur le bouton **Edit schema** dans la vue **Basic settings** de chacun des composants **tLogRow** pour avoir un aperçu des structures de données en entrée et en sortie, telles que vous les avez définies dans le **Multi Schema Editor**, ou pour les modifier.

- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

Le fichier délimité multi-schéma est lu ligne par ligne et les champs extraits sont affichés dans la console **Run Job** comme défini dans le **[Multi Schema Editor]**.

```
Starting job pivot at 10:33 18/01/2010.

01|SOFT MUSIC ALBUM|RICHARDSON|15/12/2005
02|We Danced|SOFT MUSIC ALBUM
02|She's Everything|SOFT MUSIC ALBUM
02|Once in a Lifetime Love|SOFT MUSIC ALBUM
03|National Library|SOFT MUSIC ALBUM
01|COUNTRY MUSIC ALBUM|WHITE|02/01/2006
02|Fall Into Me|COUNTRY MUSIC ALBUM
02|Another Try|COUNTRY MUSIC ALBUM
02|Something About Her|COUNTRY MUSIC ALBUM
Job pivot ended at 10:33 18/01/2010. [exit code=0]
```

tFileInputMSPositional

Propriétés du tFileInputMSPositional

Famille du composant	File/Input	
Fonction	Le composant **tFileInputMSPositional** permet de lire des schémas multiples à partir d'un fichier positionnel.	
Objectif	Le **tFileInputMSPositional** ouvre le fichier multi structure complexe, en lit la structure (schémas) puis utilise des liens de type **Row** pour envoyer les champs, tels qu'ils ont été définis dans les différents schémas, vers les composants suivants dans le Job.	
Basic settings	*Property type*	Peut être **Built-in** ou **Repository**.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Chemin d'accès et nom du fichier, et/ou variable à traiter. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Row separator	Chaîne de caractères (ex : "\n" sous Unix) séparant les lignes.
	Schema Identifier Field Position	Position de la chaîne de caractères identifiant les différents schémas.
	Records	**Schema** : définissez autant de schémas que nécessaire. **Schema Identifier value** : valeur de la chaîne de caractères identifiant les différents schémas. Saisissez le nom de la colonne. **Pattern** : Chaîne de caractères représentant la longueur de chacune des colonnes du schéma correspondant, séparées par des virgules. Vérifiez que les valeurs définies dans ce champ sont cohérentes avec le schéma défini. **Reject incorrect row size** : cochez les cases des schémas pour lesquels vous n'acceptez pas une taille de ligne incorrecte. **Parent key column** : Saisissez le nom de la colonne clé parente. **Key column** : Saisissez le nom de la colonne clé.
	Skip from header	Nombre de lignes à ignorer au début du fichier.
	Skip from footer	Nombre de lignes à ignorer à la fin du fichier.
	Limit	Nombre maximal de lignes à traiter. Si Limit = 0, aucune ligne ne sera lue ni traitée.

	Die on parse error	Décochez cette case pour passer les lignes contenant des erreurs de parse et terminer le traitement avec les lignes sans erreur.
	Die on unknown header type	Décochez cette case pour passer les lignes dont le type d'en-tête est inconnu et terminer le traitement avec les lignes sans erreurs.
Advanced settings	*Process long rows (needed for processing rows longger than 100,000 characters wide)*	Cochez cette case afin de pouvoir traiter de longues lignes (ceci est nécessaire pour traiter des lignes contenant plus de 100 000 caractères).
	Trim all columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes.
	Advanced separator (for numbers)	Cochez cette option pour modifier les séparateurs utilisés pour les nombres : **Thousands separator** : définissez le séparateur utilisé pour les milliers. **Decimal separator** : définissez le séparateur utilisé pour les décimaux.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les métadonnées de process du Job, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Utilisez ce composant pour lire un fichier positionnel multi-schéma et servez-vous d'une valeur spécifiant l'emplacement pour séparer les champs. Ce composant permet de créer un flux de données à l'aide d'un lien **Row** > **Main**, ainsi que de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données dont le type ne correspond pas au type défini. Pour un exemple d'utilisation de ces deux liens, consultez le *Scénario 2 : Extraire les données XML erronées dans un flux de rejet, page 1466* du composant **tFileInputXML**.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Lecture d'un fichier délimité multi structure, page 945* du composant **tFileInputMSDelimited**.

tFileInputMSXML

Propriétés du tFileInputMSXML

Famille du composant	XML ou File/Input	
Fonction	Le composant **tFileInputMSXML** permet de lire des schémas multiples à partir d'un fichier XML structuré et de les restituer en sortie.	
Objectif	Le **tFileInputMSXML** ouvre le fichier multi structure complexe, en lit la structure (schémas) puis utilise des liens de type **Row** pour envoyer les champs, tels qu'ils ont été définis dans les différents schémas, vers les composants suivants dans le Job.	
Basic settings	*File Name*	Chemin d'accès et nom du fichier, et/ou variable à traiter. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Root XPath query	Racine de l'arborescence XML sur laquelle se base la requête.
	Enable XPath in column "Schema XPath loop" But lose the order	Cochez cette case si vous souhaitez définir un chemin XPath dans le champ **Schema XPath loop** du tableau **Outputs**. *Cette option n'est disponible qu'avec le mode de génération* **dom4j**. *Vérifiez donc que ce mode a bien été sélectionné dans la liste* **Generation mode** *dans l'onglet* **Advanced settings** *de votre composant. Si vous utilisez cette option, les données ne seront pas retournées dans l'ordre.*
	Outputs	**Schema** : définissez autant de schémas que nécessaire. **Schema XPath loop** : noeud ou chemin XPath de l'arborescence XML sur lequel la boucle est basée. *Si vous souhaitez utiliser un chemin XPath dans le champ* **Schema XPath loop**, *vous devez préalablement cocher la case* **Enable XPath in column "Schema XPath loop" but lose the order**. **XPath Queries** : renseignez les champs à extraire de la structure XML d'entrée. **Create empty row** : cochez les cases des schémas pour lesquels vous voulez créer des lignes vides.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur.

Advanced settings	*Trim all column*	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes.
	Generation mode	Sélectionnez le mode de génération dans la liste.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les métadonnées de process du Job, aussi bien au niveau du Job qu'au niveau de chaque composant.
Limitation	n/a	

Scénario : Lecture d'un fichier XML multi structuré

Le présent scénario sert à créer un Job en Java permettant de lire un fichier XML multi-schéma et d'en afficher la structure sur la console de la vue **Run**.

Le fichier XML traité dans ce scénario se présente comme suit :

```
- <root>
  - <discotheque>
      <record>Something About Her</record>
      <record>Fall Into Me</record>
      <record>Once In A Liftime</record>
    </discotheque>
  - <bookstore>
      <book>Another Try</book>
      <book>By Myself</book>
      <book>null</book>
    </bookstore>
  </root>
```

- Cliquez et déposez le composant **tFileInputMSXML** et deux composants **tLogRow** dans l'éditeur graphique.
- Double-cliquez sur le **tFileInputMSXML** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés :

- Dans le champ **File Name**, sélectionnez le fichier XML que vous souhaitez traiter.
- Dans le champ **Root XPath query**, indiquez la racine de l'arborescence XML sur laquelle sera basée la requête.
- Cochez la case **Enable XPath in column "Schema XPath loop" But lose the order** si vous souhaitez définir un chemin XPath dans le champ **Schema XPath loop** du tableau **Outputs**. Dans ce scénario, vous n'utilisez pas cette option.
- Cliquez sur le bouton **[+]** pour ajouter des lignes au tableau **Outputs** dans lesquelles vous pouvez définir les schémas de sortie.
- Dans le tableau **Outputs**, cliquez dans chaque cellule **Schema** et cliquez sur le bouton **[...]** pour saisir le nom du schéma dans la boîte de dialogue qui s'ouvre.

- Saisissez le nom du schéma de sortie et cliquez sur **OK**.
  L'éditeur de schéma du **tFileInputMSXML** apparaît.
- Définissez le schéma déjà défini dans le tableau **Outputs**.
- Effectuez cette procédure pour chaque schéma de sortie que vous créez. Dans ce scénario, l'objectif est de définir deux schémas de sortie : *record* et *book*.
- Dans l'éditeur graphique, cliquez-droit sur le **tFileInputMSXML** et connectez-le aux **tLogRow1** et **tLogRow2** à l'aide des liens **record** et **book**.

- Dans la vue **Basic settings** et dans la cellule **Schema XPath loop**, indiquez le noeud de l'arborescence XML sur laquelle sera basée la requête.
- Dans la cellule **XPath Queries**, indiquez les champs à extraire du fichier XML d'entrée.
- Cochez les cases **Create empty row** à côté des noms des schémas dans lesquels vous souhaitez créer des lignes vides.
- Enregistrez votre Job et appuyez sur **F6** pour lancer son exécution. Les schémas spécifiés sont extraits du fichier XML multi-schéma et sont affichés dans la console.

Le fichier XML multi-schéma est lu ligne par ligne et les champs extraits sont affichés dans la console de la vue **Run**.

```
Starting job tFileInputXML_ at 15:15 21/07/2009.

.-------------------------------------------------------------.
|                          tLogRow_1                          |
|=------------------------------------------------------------=|
|recordname                                                   |
|=------------------------------------------------------------=|
|
    Something About Her
    Fall Into Me
    Once In A Liftime
  |
'-------------------------------------------------------------'

.---------------------------------------.
|                tLogRow_2              |
|=--------------------------------------=|
|bookname                               |
|=--------------------------------------=|
||
    Another Try
    By Myself
      null
  |
'---------------------------------------'
Job tFileInputXML_ ended at 15:15 21/07/2009. [exit code=0]
```

tFileInputPositional

Propriétés du tFileInputPositional

Famille de composant	File/Input	
Fonction	Le composant **tFileInputPositional** lit un fichier ou un flux de données ligne par ligne et extrait les champs selon un modèle (*pattern*).	
Objectif	**tFileInputPositional** lit un fichier ou un flux de données ligne par ligne, sépare les champs tels que définis par le schéma et passe les données extraites au composant suivant via une connnexion de type Row.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier où sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File name/Stream	**File name** : Chemin d'accès et nom du fichier, et/ou variable à traiter. **Stream** : Flux de données à traiter. Les données doivent préalablement être mises en flux afin d'être récupérées par le **tFileInputPositional** via la variable *INPUT_STREAM* disponible dans la liste d'auto-complétion (**Ctrl+Espace**). Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Row separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Customize	Cochez cette case pour personnaliser le format des données du fichier positionnel et renseignez les colonnes du tableau **Formats**. **Column** : Sélectionnez la colonne que vous souhaitez personnaliser. **Size** : Saisissez la taille correspondant à la colonne. **Padding char** : Saisissez entre guillemets le caractère de remplissage utilisé afin qu'il soit supprimé de votre champ. Le caractère par défaut est un espace. **Alignment** : Sélectionnez le paramètre d'alignement approprié.
	Pattern	Longueurs séparées par des virgules, interpétées comme une chaîne de caractères entre guillemets. Vérifiez que les valeurs saisies dans ce champ sont cohérentes avec le schéma défini.
	Skip empty rows	Cochez cette case pour ignorer les lignes vides.
	Uncompress as zip file	Cochez cette case pour décompresser le fichier d'entrée.

	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
	Header	Nombre de lignes à ignorer au début d'un fichier.
	Footer	Nombre de ligne à ignorer à la fin d'un fichier.
	Limit	Nombre maximum de lignes à traiter. Si Limit = 0, aucune ligne n'est lue ni traitée.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma sera créé et conservé pour ce composant. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Needed to process rows longer than 100 000 characters*	Cochez cette case si les lignes à traiter dans le fichier d'entrée font plus de 100 000 caractères de long.
	Advanced separator	Cochez cette case pour modifier les séparateurs de lignes utilisés.
	Trim all columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans toutes les colonnes.
	Check columns to trim	Cochez la case devant le nom de chacune des colonnes dont vous souhaitez supprimer les espaces de début et de fin de champ.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du Job, ainsi qu'au niveau de chaque composant.
Utilisation	Utilisez ce composant pour lire un fichier et séparer les champs à l'aide du séparateur spécifié. Ce composant permet de créer un flux de données à l'aide d'un lien **Row** > **Main**, ainsi que de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données dont le type ne correspond pas au type défini. Pour un exemple d'utilisation de ces deux liens, consultez le *Scénario 2 : Extraire les données XML erronées dans un flux de rejet, page 1466* du composant **tFileInputXML**.	

Scénario : Transformer un fichier positionnel en XML

Le scénario suivant construit un Job avec deux composants, qui a pour objectif de lire les données d'un fichier positionnel en entrée et de rendre des données sélectionnées en sortie (selon leur position) dans un fichier XML.

- Cliquez et déposez un composant **tFileInputPositional** de la Palette dans l'espace de modélisation. Le fichier contient des lignes de données, dans cet exemple, *contract nr*, *customer references* et *insurance numbers*.
- Cliquez-déposez un composant **tFileOutputXML**. Ce fichier recevra les références de manière structurée.
- Cliquez-droit sur le composant **tFileInputPositional** et sélectionnez une connexion *Row > Main*. Glissez cette connexion vers le composant **tFileOutputXML** et relâchez la souris lorsque le symbole de prise de courant apparaît.
- Sélectionnez le composant **tFileInputPositional** à nouveau et définissez ses propriétés.
- Les propriétés de ce Job sont de type **Built-in** pour ce scénario. Par conséquent, les informations de propriétés sont renseignées pour ce Job seulement et ne peuvent être réutilisées pour un autre Job, contrairement à des propriétés de type **Repository**.

- Renseignez le chemin d'accès au fichier dans le champ **File Name**. Ce champ est obligatoire.
- Puis définissez le séparateur de lignes (**Row separator**) permettant d'identifier la fin de la ligne : le retour chariot par défaut.
- Puis dans le champ **Pattern** définissant les champs d'une ligne. Le pattern est une série de longueurs correspondant aux valeurs de champs du fichier en entrée. Les valeurs doivent être saisies entre guillemets simples et séparées par une virgule. Veillez à ce que les valeurs saisies correspondent à la longueur des champs définis dans le schéma.
- Dans ce scénario, les champs d'en-tête (**Header**), de pied de page (**Footer**) et de limite (**Limit**) n'ont pas besoin d'être définis.

- Sélectionnez le type de schéma (**Schema**) pour définir les données à transmettre au composant **tFileOutputXML**.
- Vous pouvez sélectionner et/ou modifier le schéma via la fonction **Edit Schema**. Pour ce schéma, définissez trois colonnes, respectivement *Contracts*, *CustomerRef* et *InsuranceNr* correspondant aux trois valeurs de longueurs définies.

- Puis définissez les propriétés du composant de sortie.
- Saisissez le chemin d'accès au fichier XML de sortie.

- Saisissez une (ou plusieurs) balise racine (**Root tag**) pour envelopper la structure en sortie XML. Dans ce cas, la balise est *'ContractsList'*.
- Définissez la balise de la ligne (**Row tag**) qui définit chaque ligne. Dans ce cas, la balise est *'ContractRef'*.
- Cochez la case **Column name as tag name** pour réutiliser le libellé des colonnes du schéma d'entrée comme nom de balise pour structurer le fichier XML de sortie. Par défaut, le nom de balise est *'field'*, quel que soit le type de valeur dans les colonnes.

- Saisissez l'encodage (**Encoding**) du fichier d'entrée. Notez que pour l'instant la vérification de l'encodage n'est pas fonctionnellement supportée.
- Sélectionnez le type de schéma dans le champ **Schema**. Si vous avez déjà mis en place la connexion entre les composants d'entrée et de sortie du Job, la propagation du schéma est automatique. Sinon, cliquez sur **Sync columns** pour les synchroniser.
- Passez à la vue **Run** et cliquez sur le bouton **Run** pour éxecuter le Job.

Le fichier est lu ligne par ligne et divisé en champs basés sur les longueurs définies précédemment dans le champ **Pattern**. Vous pouvez l'ouvrir dans n'importe quel éditeur XML standard.


```
- <ContractsList>
  - <ContractRef>
      <Contract>00004</Contract>
      <CustomerRef>8200</CustomerRef>
      <InsuranceNr>50320</InsuranceNr>
    </ContractRef>
  - <ContractRef>
      <Contract>00010</Contract>
      <CustomerRef>8200</CustomerRef>
      <InsuranceNr>50335</InsuranceNr>
    </ContractRef>
  - <ContractRef>
      <Contract>00001</Contract>
      <CustomerRef>7200</CustomerRef>
      <InsuranceNr>50320</InsuranceNr>
    </ContractRef>
```

tFileInputProperties

Propriétés du tFileInputProperties

Famille de composant	File/Input	
Fonction	Le composant **tFileInputProperties** lit un fichier texte ligne par ligne et extrait les champs.	
Objectif	Il ouvre un fichier texte qu'il lit ligne par ligne, sépare les champs sur le modèle clé=valeur.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Pour ce composant, le schéma est en lecture seule puisque les fichiers sont structurés sur le modèle clé=valeur.
	File format	Sélectionnez le format de votre fichier **.properties** ou **.ini**.
		.properties : fichier de configuration dont les données sont disposées sur deux lignes et structurées de la manière suivante clé=valeur.
		.ini : fichier de configuration dont les données sont disposées sur deux lignes, structurées de la manière suivante clé=valeur.et regroupées en sections. **Retrieve Mode** : - **Retrieve All** pour récupérer l'intégralité du fichier, - **Retrieve Sections** pour récupérer les sections du fichier, - **Retrieve by section** pour récupérer les données contenues dans la section spécifiée dans le champ **Section Name**. **Section Name** : saisissez le nom de la section sur laquelle effectuer l'opération d'itération.
	File	Chemin d'accès et nom du fichier, et/ou variable à traiter. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Calculate MD5 Hash	Cochez cette case pour que l'algorithme vérifie que le fichier est bien téléchargé.
Utilisation	Utilisez ce composant pour lire un fichier et séparer les données en fonction de la structure clé=valeur du fichier.	

Scénario : Lire et mapper la clé et les valeurs de fichiers properties et alimenter un glossaire

Le scénario suivant est un Job de quatre composants lisant deux fichiers propriétés en anglais et en français. Ces données sont ensuite mises en correspondance afin de constituer un glossaire.

Les fichiers de propriétés utilisés pour ce scénario sont des fichiers contenant les chaînes de caractères utilisés pour la localisation du composant **tMysqlInput** de **Talend Open Studio**.

tMysqlInput_messages_en.properties

```
1  DBD-ODBC.INFO=Required for ODB
2  DBD-Oracle.INFO=Required for O
3  DBD-Pg.INFO=Required for Postg
4  DBD-mysql.INFO=Required for My
5  DBNAME.NAME=Database
6  DBTABLE.NAME=Table Name
7  ENABLE_STREAM.NAME=Enable stre
8  ENCODING.NAME=Encoding
9  HELP=org.talend.help.tMysqlInp
10 HOST.NAME=Host
11 LONG_NAME=Reads a MySQL table
12 NB_LINE.NAME=Number of line
13 NULL_CHAR.NAME=Null Char
14 PASS.NAME=Password
15 PORT.NAME=Port
16 PROPERTIES.NAME=Additional JDB
17 QUERY.NAME=Query
```

tMysqlInput_messages_fr.properties

```
1  DBD-ODBC.INFO=Requis pour les co
2  DBD-Oracle.INFO=Requis par Oracl
3  DBD-Pg.INFO=Requis par PostgreSQ
4  DBD-mysql.INFO=Requis par MySQL
5  DBNAME.NAME=Base de données
6  DBTABLE.NAME=Nom de table
7  ENABLE_STREAM.NAME=Activer la di
8  ENCODING.NAME=Encodage
9  HELP=org.talend.help.tMysqlInput
10 HOST.NAME=Hôte
11 LONG_NAME=Lit une table MySQL et
12 NB_LINE.NAME=Nombre de ligne
13 NULL_CHAR.NAME=Caractère Null
14 PASS.NAME=Mot de passe
15 PORT.NAME=Port
16 PROPERTIES.NAME=Paramètres JDBC
17 QUERY.NAME=Requête
```

- Cliquez-glissez deux composants **tFileInputProperties** de la famille **File** > **Input** dans le Job designer, un composant **tMap** de la famille **Processing** et un composant **tLogRow** de la famille **Log & Errors**.
- Reliez les composants entre eux via des connexions **Row** > **Main** Le second fichier, *FR*, est utilisé comme flux de référence .
- Double-cliquez sur le premier composant **tFileInputProperties** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component** :

- Dans le champ **File Format**, sélectionnez le format de votre fichier.
- Dans le champ **File Name**, cliquez sur le bouton **[...]** et sélectionnez le fichier *.properties* anglais que vous souhaitez utiliser.
- Répétez cette procédure avec le deuxième composant et le fichier français :

- Double-cliquez sur le composant **tMap** pour ouvrir son éditeur :

- Sélectionnez toutes les colonnes de la table *English_terms* et glissez-les vers la table de sortie.
- Sélectionnez la colonne *key* de la table *English_terms* et glissez-la dans la colonne *key* de la table *French_terms*.

- Dans le tableau **glossary** de la zone **Schema editor** en bas du **tMap**, renommez le champ *value* en *EN* puisqu'il contiendra les valeurs du fichier anglais.
- Cliquez sur le bouton **[+]** pour ajouter une ligne au tableau **glossary** et renommez-la **FR**.
- Dans le champ **Length**, configurez la taille maximum du champ à *255*.
- Dans le coin supérieur gauche de l'éditeur du **tMap**, sélectionnez la colone value de la table *English_terms* et glissez-la dans la colonne *FR* de la table *French_terms*.
- Cliquez sur **OK** pour valider vos changements et fermer l'éditeur.
- Dans le Job designer, double-cliquez sur le composant **tLogRow** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component**.
- Cliquez sur le bouton **Sync Columns** pour récupérer le schéma du composant précédent.
- Enregitrez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job tFileInputProperties at 15:25 29/05/2009.

PORT.NAME | Port | Port
HELP | org.talend.help.tMysqlInput | org.talend.help.tMysqlInput
STRING_QUOTE.NAME | String Quote | Séparateur de chaine de
caractère
DBTABLE.NAME | Table Name | Nom de table
QUERYSTORE.NAME | Query Type | Type de requête
SQL_SYNTAX.NAME | Sql Syntax | Syntaxe SQL
TYPE.ITEM.PGSQL | PostgreSQL | PostgreSQL
TYPE.NAME | Database Driver | Pilote de base de données
TABLE.NAME | Table Name | Nom de table
ENCODING.NAME | Encoding | Encodage
QUERY.NAME | Query | Requête
DBD-ODBC.INFO | Required for ODBC-like connection | Requis pour
les connexions de type ODBC
```

Le glossaire apparaît dans la console de la vue **Run** avec dans la première colonne le nom de la clé, dans la deuxième le terme du fichier anglais et dans le troisième colonne le terme correspondant du fichier français.

tFileInputRegex

Propriétés du tFileInputRegex

Famille de composant	File/Input	
Fonction	Puissant composant qui peut remplacer bon nombre des composants File. Il requiert des connaissances avancées en rédaction d'expressions régulières.	
Objectif	Le **tFileInputRegex** lit un fichier ou un flux de données ligne par ligne afin de le diviser en champs à l'aide d'expressions régulières et d'envoyer les champs définis dans le schéma au composant suivant via une connexion Row.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File name/Stream	**File name** : Chemin d'accès et nom du fichier, et/ou variable à traiter. **Stream** : Flux de données à traiter. Les données doivent préalablement être mises en flux afin d'être récupérées par le **tFileInputRegex** via la variable *INPUT_STREAM* disponible dans la liste d'auto-complétion (**Ctrl+Espace**). Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Row separator	Chaîne (ex : "\n" sous Unix) pour distinguer les lignes.
	Regex	Ce champ est compatible Perl ou Java et peut contenir plusieurs lignes. Intégrez à vos expressions régulières le subpattern correspondant aux champs à extraire. En Java, doublez les antislashes en regexp *La syntaxe Regexp est différente en Java et en Perl et requiert des guillemets respectivement doubles et simples.*
	Header	Nombre de lignes à ignorer en début de fichier.
	Footer	Nombre de lignes à ignorer en fin de fichier.
	Limit	Nombre maximum de lignes à traiter. Si Limit = 0, aucune ligne n'est lue ni traitée.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.

		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et stocké dans le Repository. Ainsi, il est réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Skip empty rows	Cochez cette case pour ignorer les lignes vides.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
Utilisation	Ce composant sert à lire un fichier et à en séparer les champs selon les Regex définies.Ce composant permet de créer un flux de données à l'aide d'un lien **Row** > **Main**, ainsi que de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données dont le type ne correspond pas au type défini. Pour un exemple d'utilisation de ces deux liens, consultez le *Scénario 2 : Extraire les données XML erronées dans un flux de rejet, page 1466* du composant **tFileInputXML**.	
Limitation	n/a	

Scénario : Transformer en fichier Regex en Positional

Le scénario suivant est un Job formé de deux composants, qui a pour objectif de lire les données d'un fichier d'entrée utilisant des expressions régulières et transformant les données ainsi délimitées en sortie positionnelle.

- Cliquez et déposez un composant **tFileInputRegex** et **tFileOutputPositional** de la **Palette** dans l'espace de modélisation.
- Cliquez droit sur le composant **tFileInputRegex** et sélectionnez une connexion **Row** > **Main** dans le menu contextuel. Faites glisser cette connexion vers le composant **tFileOutputPositional** et relâchez lorsque le symbole approprié apparaît.
- Sélectionnez **tFileInputRegex** à nouveau et définissez ses propriétés dans l'onglet **Component** :

- Les propriétés de ce Job sont de type built-in. Par conséquent, elles ne sont définies que pour ce Job et ne peuvent être partagées avec d'autres utilisateurs.
- Indiquez le chemin d'accès au fichier dans le champ **File Name**. Ce champ est obligatoire.
- Définissez le séparateur de lignes, dans le champ **Row separator**, afin d'identifer la fin d'une ligne.
- Puis, dans **Regular expression**, définissez l'expression régulière à utiliser pour délimiter les champs d'une ligne qui seront passés au composant suivant. Vous pouvez utiliser Java ou Perl et saisir une expression régulière multiligne si vous en avez besoin.

Veillez à utiliser la bonne expression régulière en fonction de la langue de génération de code que vous utilisez. En effet, la syntaxe n'est pas la même en Java et en Perl. Assurez-vous aussi d'insérer les expressions régulières dans des guillemets simples ou doubles en fonction de la langue de génération.

- Assurez-vous d'inclure dans cette expression tous les sous-patterns des champs à extraire.
- Dans ce scénario, ignorez les champs en-tête (**Header**), pied de page (**Footer**) et limite (**Limit**).
- Dans le **Schema**, sélectionnez local (**Built-in**) comme type de schéma des données à transférer au composant **tFileOutputPositional**.
- Vous pouvez charger ou créer à partir de la fonction **Edit Schema**.
- Puis définissez les propriétés du deuxième composant :

- Saisissez le chemin d'accès au fichier de sortie.
- Saisissez l'encodage du fichier de sortie dans le champ **Encoding**. Notez que pour l'instant, l'homogénéité de l'encodage dans un Job n'est pas vérifiée.
- Sélectionnez le type de schéma dans **Schema**. Cliquez sur le bouton **Sync columns** pour récupérer en sortie le schéma d'entrée.
- Passez à l'onglet **Run** et cliquez sur le bouton **Run** pour exécuter le Job.

Le fichier d'entrée est lu ligne par ligne puis divisé en champs délimités selon la valeur de l'expression regulière définie. Vous pouvez ouvrir le fichier de sortie avec tout éditeur standard.

tFileInputXML

Le **tFileInputXML** appartient à deux familles de composants : File et XML. Pour plus d'informations concernant le composant **tFileInputXML**, consultez *tFileInputXML, page 1462*.

tFileList

Propriétés du tFileList

Famille de composant	File/Management	
Fonction	Le composant **tFileList** fait une boucle sur les fichiers et/ou les dossiers d'un répertoire donné.	
Objectif	Le **tFileList** récupère un groupe de fichiers et/ou de dossiers suivant un masque donné et fait une boucle sur chacun d'eux.	
Basic settings	*Directory*	Chemin d'accès au répertoire de fichiers.
	FileList Type	Sélectionnez à partir de la liste déroulante le type de source d'entrée utilisé : **Files** pour des fichiers, **Directories** pour des répertoires ou **Both** pour les deux types.
	Include subdirectories	Cochez cette case si le ou les répertoires sélectionné(s) contiennent des sous-dossiers.
	Case Sensitive	Cochez cette case pour créer un filtre selon la casse des noms des fichiers.
	Generate Error if no file found	Cochez cette case pour être averti par un message d'erreur que la recherche de fichier n'a pas abouti.
	Use Glob Expressions as Filemask (Unchecked means Perl5 Regex Expressions)	Cette case est cochée par défaut et permet de filtrer les résultats à l'aide d'expressions globales (**Glob Expressions**). Décochez la case pour filtrer les résultats à l'aide d'expressions de type regex Perl5.
	Files	Cliquez sur le bouton **[+]** pour ajouter des lignes de filtre : **Filemask** : entrez un nom de fichier ou masque de fichier utilisant un caractère spécial ou une expression régulière.

	Order by	Les dossiers sont listés en premier, puis viennent ensuite les fichiers. Vous pouvez définir les priorités des dossiers et des fichiers de différentes manières : **By default** : par ordre alphabétique, par dossier puis par fichier, **By file name** :par ordre alphabétique ou ordre alphabétique inversé, **By file size** : du plus petit léger au plus lourd ou du plus lourd au plus léger, **By modified date** : du plus récent au plus ancien ou du plus ancien au plus récent. Lorsque vous sélectionnez l'option **By file name**, si deux fichiers ont le même nom, le classement s'effectuera via l'option **By modified date**. Si vous choisissez l'option **By file size**, et que les fichiers font le même poids, ils seront classés selon l'option **By file name**. Si vous sélectionnez l'option **By modified date** et que la date de modification est la même pour deux fichiers, alors ils seront classés selon l'option **By file name**.
	Order action	Choisissez de classer par ordre : **ASC** : ordre alphabétique / du plus petit au plus grand / du plus ancien au plus récent. **DESC** : à l'inverse de l'ordre alphabétique / du plus grand au plus petit / du plus ancien au plus récent.
Utilisation	Le composant **tFileList** fournit une liste de fichiers d'un répertoire donné sur lequel il fait une boucle.	

Global Variables		**Current File Name** : indique le nom actuel du fichier. Disponible comme variable **Flow**. Retourne une chaîne de caractères. **Current File Name with Path** : indique le nom actuel du fichier et son chemin d'accès. Disponible comme variable **Flow**. Retourne une chaîne de caractères. **Current File Extension** : indique l'extension actuelle du fichier. Disponible comme variable **Flow**. Retourne une chaîne de caractères. **Current File Directory** : indique le chemin d'accès au dossier ou au sous-dossier dans lequel le fichier est stocké. Disponible comme variable **Flow**. Retourne une chaîne de caractères. **Number of files** : indique le nombre de fichiers sur lesquels a été effectuée une boucle. Disponible comme variable **Flow**. Retourne un entier. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Iterate. **Trigger** : OnSubjobOk, OnSubjobError, OnComponentOk, OnComponentError. Les liens d'entrée (d'un composant à un autre) : **Row** : Iterate. **Trigger** : Run If, OnSubjobOk, OnSubjobError, Synchronize, Parallelize. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** de **Talend Open Studio**.

Scénario : Itération sur un répertoire

Le scénario suivant décrit un Job de trois composants, qui a pour objectif de répertorier les fichiers d'un répertoire donné, de lire chaque fichier par itération, de sélectionner les données délimitées et d'afficher ces données en sortie dans la console **Run**.

- Cliquez et déposez les composants suivants : **tFileList**, **tFileInputDelimited** et **tLogRow** dans l'espace de modélisation.
- Cliquez-droit sur le composant **tFileList** et connectez-le au composant **tFileInputDelimited** via une connexion de type **Row Iterate**. Puis connectez le **tFileInputDelimited** au **tLogRow** à l'aide d'une connexion de type **Row Main**.
- Paramétrez ensuite les propriétés respectives des composants.
- Tout d'abord, sélectionnez le composant **tFileList** et cliquez sur l'onglet **Basic settings** :

- Dans le champ **Directory**, indiquez le chemin d'accès des fichiers à traiter. Pour faire apparaître ce chemin d'accès dans le Job lui-même, utilisez le nom de champ (__DIRECTORY__) qui apparaît en bulle info lorsque vous passez le souris sur le champ **Directory**. Saisissez cette référence dans le champ **Label Format** de l'onglet **View**.

- Dans le champ **FileList Type**, sélectionnez l'option **Files**.
- Définissez le champ **Case sensitive** sur **Yes** pour prendre la casse en compte dans cet exemple.
- Laissez la case **Use Glob Expressions as Filemask** cochée si vous utilisez des expressions globales pour filtrer les fichiers.
- Dans le champ **Filemask**, indiquez un masque de fichier. Utilisez les caractères spéciaux si besoin est.

- Paramétrez les propriétés du deuxième composant **tFileInputDelimited** :

- Dans le champ **File Name**, nommez le fichier à l'aide d'une variable contenant son chemin d'accès, tel que vous l'avez renseigné dans les propriétés du **tFileList**. Appuyez sur **Ctrl+Espace** pour accéder à la liste d'autocomplétion des variables.
- Sélectionnez la variable globale `((String)globalMap.get("tFileList_1_CURRENT_FILEPATH"))` si vous travaillez en Java, ou `$_globals{tFileList_1}{CURRENT_FILEPATH}` si vous travaillez en Perl. De cette manière, tous les fichiers du répertoire source peuvent être traités.
- Renseignez les autres champs comme pour un fichier délimité **tFileInputDelimited** standard. Voir également : *Propriétés du tFileInputDelimited, page 915*.
- Sélectionnez le composant de sortie **tLogRow** et renseignez le séparateur de champs qui s'affichera dans la console de l'onglet **Run**. Voir également : *Propriétés du tLogRow, page 1163*.

```
Starting job test at 17:16 21/09/2009.

.--+-----------------------------+-----------------------+-------.
|                          tLogRow_1                                |
|=-+-----------------------------+-----------------------+------=|
|id|CustomerName                 |CustomerAddress        |idState|
|=-+-----------------------------+-----------------------+------=|
|1 |Griffith Paving and Sealcoatin|talend@apres91        |7      |
|2 |Bill's Dive Shop             |511 Maple Ave.  Apt. 1B|35     |
|3 |Childress Child Day Care     |662 Lyons Circle       |1      |
|4 |Facelift Kitchen and Bath    |220 Vine Ave.          |41     |
|5 |Terrinni & Son Auto and Truck|770 Exmoor Rd.         |5      |
|6 |Kermit the Pet Shop          |1860 Parkside Ln.      |28     |
|7 |Tub's Furniture Store        |807 Old Trail Rd.      |15     |
|8 |Toggle & Myerson Ltd         |618 Sheriden rd.       |9      |
|9 |Childress Child Day Care     |788 Tennyson Ave.      |12     |
|10|Elle Hypnosis and Therapy Cent|2032 Northbrook Ct.    |1      |
'--+-----------------------------+-----------------------+-------'
.--+-----------------------------+-----------------------+-------.
|                          tLogRow_1                                |
|=-+-----------------------------+-----------------------+------=|
|id|CustomerName                 |CustomerAddress        |idState|
|=-+-----------------------------+-----------------------+------=|
|1 |Glenwood Credit Union        |511 Maple Ave.  Apt. 1B|46     |
|2 |Gourmet the Frog             |788 Tennyson Ave.      |1      |
|3 |Acturial Enterprises Ltd.    |3385 University Ave.   |34     |
|4 |Salt & Pepper Catering Service|965 Marion Place Apt. 65C|44     |
|5 |Rythmics Ltd.                |1875 Roger Williams Ave.|22     |
|6 |Parkway Auto Body            |1859 Green Bay Rd.     |27     |
|7 |Futoons Cartoons Emporium    |1486 Old Deerfield Rd. |24     |
|8 |Glenn Oaks Office Supplies   |1882 St. Johns         |10     |
|9 |Garfield Appliance Service   |3150 Skokie Valley Rd. |33     |
|10|New Dehli Auto Exchange      |1957 Huntington Ave.   |49     |
'--+-----------------------------+-----------------------+-------'
Job test ended at 17:16 21/09/2009. [exit code=0]
```

L'itération s'effectue sur le répertoire défini. Chacun des fichiers contenus dans ce répertoire est lu. Puis les données délimitées sont transmises au composant de sortie qui les affiche dans la console Log de la vue **Run**.

Pour d'autres scénarios utilisant le **tFileList**, consultez *tFileCopy, page 900*

tFileOutputARFF

Propriétés du tFileOutputARFF

Famille de composant	File/Output	
Fonction	Le composant **tFileOutputARFF** transmet des données vers un fichier ARFF.	
Objectif	Ce composant écrit un fichier ARFF contenant des données organisées en fonction du schéma défini.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant connexion de base de données et enregistrer les paramètres de connexion que vous avez définis dans la vue **Basic settings** du composant. Pour plus d'informations sur comment définir et stocker des paramètres de connexion de base de données, consultez le chapitre *Configurer une connexion DB* du **Guide Utilisateur** de **Talend Open Studio** .
	File name	Chemin d'accès et nom du fichier de sortie, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Attribute Define	Reflète le schéma tel qu'il est défini dans la boîte de dialogue **Edit schema**. **Column** : Nom de la colonne. **Type** : Type des données de la colonne. **Pattern** : Saisissez le modèle (pattern) de la donnée si nécessaire.
	Relation	Saisissez le nom de la relation.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le Repository.

		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Create directory if not exists	Cette case est cochée par défaut. Cette option permet de créer le dossier contenant le fichier de sortie s'il n'existe pas déjà.
Advanced settings	*Don't generate empty file*	Cochez cette case pour empêcher la génération d'un fichier vide.
	tStatCatcher Statistics	Cochez cette case pour collecter les métadonnées de process du Job, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Utilisez ce composant pour écrire dans un fichier ARFF des données récupérées d'autres composants à l'aide d'une connexion de type **Row**.	
Global Variables		Les variables globales peuvent être utilisées comme paramètres dans la plupart des champs de propriétés descurseur dans le champ puis appuyez sur **Ctrl+Espace**. Double-cliquez sur la variable afin de populer le champ. La principale variable globale associée au **tFileOutputARFF** est : **Number of Lines** : indique le nombre de lignes traitées. Disponible comme variable **After**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main. **Trigger** : OnSubjobOk, OnSubjobError, Run if. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject, Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Afficher le contenu d'un fichier ARFF, page 912*.

tFileOutputDelimited

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tFileOutputDelimited

Famille de composant	File/Output	
Fonction	Le composant **tFileOutputDelimited** transmet des données vers un fichier délimité.	
Objectif	Ce composant écrit un fichier délimité contenant des données organisées en fonction du schéma défini.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File name	Chemin d'accès et nom du fichier de sortie, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Row separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Field separator	Caractère, chaîne ou expression régulière pour séparer les champs du fichier de sortie.
	Append	Cochez cette option pour ajouter de nouvelles lignes à la fin du fichier.
	Include header	Cochez cette case pour tenir compte de l'en-tête dans le fichier de sortie.
	Compress as zip file	Cochez cette case pour compresser en zip le fichier de sortie.

	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le Repository.. Si vous effectuez des modifications, le schéma passe automatiquement en mode **Built-In**. Ce composant possède l'avantage d'avoir la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues à partir de fichiers sources ou de copier des colonnes par lots à partir d'un fichier source, sans avoir à les mapper individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez la section *Onglet Dynamic settings* dans le **Guide Utilisateur Talend Integration Suite Studio**.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Sync columns	Cliquez sur ce bouton pour synchroniser le schéma de sortie avec celui d'entrée. La fonction **Sync** ne s'affiche que si une connexion de type **Row** est liée au composant de sortie.
Advanced settings	*Advanced separator (for numbers)*	Cochez cette option pour modifier les séparateurs utilisés pour les nombres : **Thousands separator** : définissez le séparateur utilisé pour les milliers. **Decimal separator** : définissez le séparateur utilisé pour les décimaux.
	CSV options	Cochez cette case pour prendre en compte les paramètres spécifiques aux fichiers CSV, notamment la manière de protéger les caractères dans le champ **Escape char** et le type de guillemet dans le champ **Enclosure**.
	Create directory if not exists	Cette case est cochée par défaut. Cette option permet de créer le dossier contenant le fichier de sortie s'il n'existe pas déjà.
	Split output in several files	Si le fichier délimité en sortie est volumineux, vous pouvez le scinder en plusieurs fichiers. **Rows in each output file** : saisissez le nombre de lignes pour chaque fichier de sortie.
	Custom the flush buffer size	Cochez cette case pour personnaliser le nombre de lignes à écrire avant de vider le tampon. **Row Number**: Nombre de lignes à écrire avant de vider le tampon.

	Output in row mode	Cochez cette case pour que l'écriture se fasse en mode ligne.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	Don't generate empty file	Cochez cette case pour empêcher la génération d'un fichier vide.
	tStatCatcher Statistics	Cochez cette case pour collecter les métadonnées de process du Job, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Utilisez ce composant pour écrire dans un fichier délimité des données récupérées d'autres composants à l'aide d'une connexion de type **Row**.	
Limitation	n/a	

Scénario : Ecrire des données dans un fichier délimité

Ce scénario est constitué de trois composants et a pour objectif d'extraire certaines données d'un fichier contenant des informations sur les clients (*customers*) puis de les écrire dans un fichier délimité en sortie.

Dans l'exemple suivant, vous avez enregistré les informations de connexion et les métadonnées dans le **Repository** afin de faciliter leur réutilisation. Pour plus d'informations sur comment centraliser les données de connexion et autres métadonnées d'un fichier dans le **Repository**, consultez *Configurer un schéma File Delimited, page 304*.

- Dans le **Repository**, ouvrez le noeud **File Delimited** de la catégorie **Metadata**. Cliquez sur la métadonnée *customers*, et déposez-le dans l'espace de modélisation graphique. Une boîte de dialogue s'ouvre et vous demande quel type de composant vous voulez déposer.

- Cliquez sur **tFileInputDelimited** puis sur **OK** Un composant **tFileInputDelimited** contenant le nom de votre schéma d'entrée s'affiche dans l'espace de modélisation graphique.
- Déposez un composant **tMap** et un composant **tFileOutputDelimited** de la **Palette** dans l'espace de modélisation graphique.
- Reliez les composants à l'aide d'un lien **Row** > **Main**.
- Double-cliquez sur le composant afin d'afficher la vue **Component**, afin de paramétrer les informations de connexion et définir le schéma des données d'entrée.

- Si vous sélectionnez **Repository** dans la liste **Property Type**, les champs suivants seront automatiquement renseignés à partir des informations centralisées dans le **Repository**. Si vous sélectionnez le mode **Built-In**, renseignez les champs manuellement.
- Cliquez sur le bouton **[...]** correspondant au champ **File Name**. Dans la boîte de dialogue de sélection, parcourez le système de fichiers et sélectionnez le fichier d'entrée, dans cet exemple: *customer.csv*.

- Dans les champs **Row** et **Field Separators**, saisissez respectivement le séparateur de lignes "*\n*" et le séparateur de champs "*;*".
- Si votre fichier contient des en-têtes ou des pieds-de-page, indiquez le nombre de lignes concernées dans les champs **Header** et **Footer**. Dans cet exemple, saisissez *6* dans le champ **Header**. Vous pouvez également définir une limite au nombre de lignes traitées, en saisissant un nombre dans le champ **Limit**.
- Si vous avez sélectionné le mode **Repository**, le schéma du fichier sera automatiquement renseigné. En mode **Built-in**, cliquez sur **Edit Schema** afin de décrire les données d'entrée, et ainsi définir le schéma traité.

Le schéma du fichier *Customers* se présente comme suit :

- Cliquez sur **OK** pour fermer la boîte de dialogue.
- Double-cliquez sur le **tMap** afin d'ouvrir son éditeur.

- Dans cet éditeur, cliquez sur le bouton **plus** vert en haut à droite pour ouvrir la boîte de dialogue **[Add a new output table]**.

- Saisissez un nom pour la table que vous souhaitez créer, *row2* dans cet exemple.
- Cliquez sur **OK** pour valider vos modifications et fermer la boîte de dialogue.
- Dans la table de gauche (*row1*), sélectionnez les trois premières lignes (*Id*, *CustomerName* et *CustomerAddress*) en appuyant sur **Ctrl**, puis glissez-les dans la table de droite.
- Dans le **Schema editor** en bas de l'éditeur, changez le **Type** de la ligne **RegisterTime**, dans la table de droite, et passez-le en chaîne de caractères, **String**.

Column	Key	Type	N..	Date Pattern ...	Length	Precision	Def...	Comm...
idState		int			2			
id2		int			2			
RegTime		String			30			
RegisterTime		String			30			
Sum1		float			10	5		
Sum2		float			10	5		

- Cliquez sur **OK** pour enregistrer les modifications et fermer l'éditeur.
- Dans l'espace de modélisation, double-cliquez sur le composant **tFileOutputDelimited** pour ouvrir la vue **Component** et définir ses propriétés.

- Dans le champ **Property Type**, sélectionnez le mode **Built-in** et renseignez les champs suivants manuellement.
- Cliquez sur le bouton **[...]** correspondant au champ **File Name**. Dans la boîte de dialogue de sélection, indiquez le répertoire de destination et le nom de votre fichier.
- Pour les champs **Row** et **Field Separators**, saisissez respectivement le séparateur de lignes "*\n*" et le séparateur de champs "*;*". Ils seront utilisés dans le fichier de sortie pour délimiter les valeurs et les lignes.

- Cochez la case **Include Header** si vous souhaitez récupérer le libellé des colonnes.
- Cliquez sur **Edit schema** et assurez-vous que le schéma est correctement récupéré du schéma d'entrée. Si nécessaire, cliquez sur **Sync Columns** pour récupérer le schéma du composant précédent.
- Pour exécuter le job, appuyez sur **F6**.

customerselection.txt

```
1  id;CustomerName;CustomerAddress
2  1;Griffith Paving and Sealcoatin;talend@apres91
3  2;Bill's Dive Shop;511 Maple Ave.  Apt. 1B
4  3;Childress Child Day Care;662 Lyons Circle
5  4;Facelift Kitchen and Bath;unknown
6  5;Terrinni & Son Auto and Truck;770 Exmoor Rd.
7  6;Kermit the Pet Shop;1860 Parkside Ln.
8  7;Tub's Furniture Store;807 Old Trail Rd.
9  8;Toggle & Myerson Ltd;618 Sheriden rd.
10 9;Childress Child Day Care;788 Tennyson Ave.
11 10;Elle Hypnosis and Therapy Cent;2032 Northbrook Ct.
```

Le fichier créé est délimité par des points-virgules, il ne contient que les colonnes *Id*, *CustomerName* et *CustomerAddress*.

Pour un exemple d'utilisation du schéma dynamique dans le **tFileOutputDelimited**, consultez :

- *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507*

tFileOutputEBCDIC

Ce composant requiert l'utilisation d'un JDK Sun.

Propriétés du tFileOutputEBCDIC

Famille de composant	File/Output	
Fonction	Le composant **tFileOutputEBCDIC** écrit un fichier EBCDIC à partir de plusieurs fichiers de données différents ayant chacun un schéma différent.	
Objectif	Ce composant écrit un fichier EBCDIC avec des données extraites de fichier(s) en fonction des schémas décrivant leur structure.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Chemin d'accès et nom du fichier EBCDIC de sortie, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Xc2j file	Sélectionnez le fichier xc2j de transformation.
Utilisation	Utilisez ce composant pour écrire un fichier EBCDIC et séparer les données en fonction des schémas identifiés dans les fichiers en entrée.	

Scénario : Créer un fichier EBCDIC à partir de deux fichiers délimités

Ce scénario utilise l'assistant **[Copybook Connection]** guidant l'utilisateur à travers les différentes étapes de création de connexion au fichier Copybook et de récupération des schémas EBCDIC. L'assistant **[Copybook Connection]** est uniquement disponible pour les utilisateurs **Talend Integration Suite**. Si vous êtes utilisateur de **Talend Open Studio** ou **Talend On Demand**, vous devez configurer les propriétés du composant **tFileOutputEBCDIC** manuellement.

Le scénario suivant est un Job de trois composants ayant pour objectif d'écrire un fichier au format EBCDIC à partir de deux fichiers délimités ayant des schémas différents.

Ce scénario Java utilise l'assistant **EBCDIC Connection** pour créer une connexion au fichier Copybook et générer un fichier xc2j permettant de récupérer et de transformer les différents schémas du fichier.

- Créez une connexion au fichier Copybook décrivant la structure d'un fichier EBCDIC. Dans ce scénario la connexion Copybook est appelée *EBCDIC*.
- Récupérez le schéma du fichier.

Une fois la connexion au fichier Copybook créée et les schémas récupérés à l'aide des assistants EBCDIC et Schema, ils apparaissent sous le noeud **Metadata** > **Copybook**. Ces nouveaux schémas s'appellent *Schema01*, *Schema04* et *Schema05*.

Pour créer un fichier EBCDIC à partir de deux fichiers délimités dans **Talend Open Studio** :

- Cliquez et déposez deux composants **tFileInputDelimited** et un composant **tFileOutputEBCDIC** dans l'éditeur graphique.
- Pour les relier, cliquez-droit sur chaque composant **tFileInputDelimited**, sélectionnez **Row** > **Main** dans le menu contextuel et cliquez sur le composant **tFileOutputEBCDIC**.
- Double-cliquez sur le premier **tFileInputDelimited** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés :

- Dans le champ **File Name**, sélectionnez le fichier délimité en cliquant sur le bouton **[...]**.
- Dans le champ **Schema**, sélectionnez l'option **Repository**, puis cliquez sur le bouton **[...]** et sélectionnez le schéma correspondant à votre fichier dans la boîte de dialogue, sous le noeud **Copybook**.

- Dans le champ **Header**, indiquez le nombre de ligne en-tête à récupérer et passer au composant suivant : *1* dans ce scénario.
- Paramétrez les propriétés du deuxième **tFileInputDelimited** de la même manière que le premier.
- Double-cliquez sur le **tFileOutputEBCDIC** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés :

- Dans le champ **Data file**, indiquez le répertoire et le nom du fichier EBCDIC qui sera créé à partir des deux fichiers délimités.
- Dans le champ **Xc2j file**, renseignez le chemin d'accès au fichier permettant d'extraire le schéma décrivant la structure du fichier EBCDIC.
- Enregistrez votre Job via **Ctrl+S** et cliquez sur la vue **Run**, cochez les cases **Statistics** et **Exec time** puis cliquez sur **Run** pour exécuter le Job

tFileOutputExcel

Propriétés du tFileOutputExcel

Famille de composant	File/Output	
Fonction	Le composant **tFileOutputExcel** transmet des données vers un fichier de type MS Excel.	
Objectif	Ce composant écrit un fichier MS Excel contenant des données organisées en fonction du schéma défini.	
Basic settings	*File name*	Chemin d'accès et nom du fichier de sortie, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Sheet name	Nom de la feuille.
	Include header	Cochez cette case pour tenir compte de l'en-tête dans le fichier de sortie.
	Append existing file	Cochez cette option pour ajouter de nouvelles lignes à la fin du fichier. **Append existing sheet** : Cochez cette option pour ajouter de nouvelles lignes à la fin de la feuille Excel.
	Is absolute Y pos.	Cochez cette case pour rajouter des informations dans des cellules préexistantes : **First cell X** : emplacement de la cellule sur l'axe des abcisses. **First cell Y** : emplacement de la cellule sur l'axe des ordonnées. **Keep existing cell format** : lorsque la case est cochée, cette option permet de garder le format et l'alignement de la cellule.
	Font	Sélectionnez dans la liste déroulante la police à utiliser.
	Define all columns auto size	Cette case est décochée par défaut, vous pouvez alors préciser dans le tableau associé quelles sont les colonnes à définir automatiquement. Cochez cette case pour que la taille de toutes les colonnes soit définies automatiquement.
	Schema et Edit Schema	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (Built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.

		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Sync columns	Cliquez sur ce bouton pour synchroniser le schéma de sortie avec celui d'entrée. La fonction **Sync** ne s'affiche que si une connexion de type **Row** est liée au composant de sortie.
Advanced settings	*Create directory if not exists*	Cette case est cochée par défaut. Cette option permet de créer le dossier contenant le fichier de sortie s'il n'existe pas déjà.
	Advanced separator (for numbers)	Cochez cette option pour modifier les séparateurs utilisés pour les nombres : **Thousands separator** : définissez le séparateur utilisé pour les milliers. **Decimal separator** : définissez le séparateur utilisé pour les décimaux.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les métadonnées de process du Job, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Utilisez ce composant pour écrire dans un fichier MS Excel des données récupérées d'autres composants à l'aide d'une connexion de type **Row**.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario: Extraire les données d'un compte à partir de SugarCRM, page cxxvi*.

tFileOutputJSON

Propiétés du tFileOutputJSON

Famille de composant	File	
Fonction	Le composant **tFileOutputJSON** écrit des données dans un fichier structuré JSON de sortie.	
Objectif	Ce composant reçoit des données et les réécrit dans un bloc de données structurées JSON, dans un ficher de sortie.	
Basic settings	*File Name*	Nom du fichier de sortie et son chemin d'accès.
	Name of data blocks	Saisissez, entre guillemets doubles, un nom pour le bloc de données à écrire.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Sync columns	Cliquez pour synchroniser le schéma du fichier de sortie et le schéma du fichier d'entrée. La fonction Sync ne s'affiche que lorsqu'un lien Row connecte le composant de sortie.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du Job, ainsi qu'au niveau de chaque composant.
Utilisation	Utilisez ce composant pour réécrire des données reçues dans un fichier de sortie structuré JSON.	
Limitation	n/a	

Scénario : Ecrire un fichier JSON structuré

Ce scénario de deux composants comprend un composant **tRowGenerator** qui génère des données au hasard, ainsi qu'un composant **tFileOutputJSON** qui écrit dans un fichier structuré JSON de sortie.

- Glissez de la **Palette** dans l'espace de modélisation graphique un **tRowGenerator** et un **tFileOutputJSON**.
- Liez les composants à l'aide d'un lien **Row** > **Main**.
- Double-cliquez sur le **tRowGenerator** pour définir ses **Basic Settings** dans la vue **Component**.

- Si le schéma dont vous avez besoin est déjà stocké sous le noeud **Db Connections** dans le noeud **Repository**, sélectionnez **Repository** dans le champ **Schema** et choisissez les métadonné dans la liste.
- Sinon, cliquez sur **[...]** à côté du bouton **Edit Schema** afin d'afficher la boîte de dialogue et définir son schéma.

- Cliquez sur le bouton **[+]** pour ajouter les colonns désirées.
- Under **Columns** type in the column names.
- Sous **Type**, sélectionnez le type dans la liste.
- Cliquez sur **OK** pour fermer la boîte de dialogue
- Cliquez sur le bouton **[+]** à côté de **RowGenerator Editor** pour ouvrir la boîte de dialogue correspondante.

- Sous **Functions**, sélectionnez les fonctions pré-définies pour les colonnes, si besoin, ou sélectionnez **[...]** pour définir les paramètres personnalisés des fonctions, dans l'onglet **Function parameters**.
- Saisissez le nombre de lignes à générer, dans le champ **Number of Rows for RowGenerator**.
- Cliquez sur **OK** pour fermer la boîte de dialogue.
- Cliquez sur le composant **tFileOutputJSON** afin de définir ses propriétés, dans l'onglet **Basic Settings** de la vue **Component**.

- Cliquez sur le bouton **[...]** pour parcourir votre répertoire et définir où vous souhaitez générer le fichier de sortie JSON, puis saisissez un nom de fichier.
- Saisissez un nom pour le bloc de données à générer dans le champ correspondant, entre guillemets doubles.
- Sélectionnez **Built-In** dans la liste **Schema**.
- Cliquez sur **Sync Columns** afin de retrouver le schéma du précédent composant.
- Appuyez sur **F6** pour exécuter le Job.

```
{
    "person":[
        {
            "birthplace":"Manchester",
            "age":48,
            "name":"Carter"
        },
        {
            "birthplace":"Liverpool",
            "age":39,
            "name":"Clinton"
        },
        {
            "birthplace":"London",
            "age":53,
            "name":"Taylor"
        }
]
}
```


Les données du schéma d’entrée sont écrites dans un bloc de données structurées JSON, dans le fichier de sortie.

tFileOutputLDIF

Propriétés du tFileOutputLDIF

Famille de composant	File/Output	
Fonction	Le composant **tFileOutputLDIF** extrait des données vers un fichier LDIF qui peut ensuite être chargé dans un annuaire LDAP.	
Objectif	Le **tFileOutputLDIF** écrit ou modifie un fichier LDIF avec des données délimitées en entrée selon le schéma défini. Peut également supprimer le contenu d'un fichier LDIF.	
Basic settings	*File name*	Chemin d'accès et nom du fichier de sortie, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Wrap	Place un retour chariot après le nombre de caractères défini.
	Change type	Sélectionnez **Add**, **Modify** ou **Delete** pour respectivement créer un nouveau fichier LDIF, modifier ou supprimer un fichier LDIF. En cas de modification, paramétrez le type de modification apportée.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Sync columns	Cliquez pour synchroniser le schéma du fichier de sortie avec le schéma du fichier d'entrée. La fonction Sync n'apparaît qu'une fois la connexion Row mise en place entre les composants d'entrée et de sortie.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.

Advanced settings	*Advanced separator*	Cochez cette case pour modifier les séparateurs de lignes utilisés.
	Create directory if not exists	Cochez cette case pour créer un nouveau dossier si votre dossier de destination n'existe pas.
	Custom the flush buffer size	Cochez cette case pour personnaliser le nombre de lignes à écrire avant de vider le tampon. **Row Number**: Nombre de lignes à écrire avant de vider le tampon.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du Job, ainsi qu'au niveau de chaque composant.
Utilisation	Utilisez ce composant pour écrire un fichier LDIF à l'aide des données provenant d'autres composants reliés par une connexion de type Row.	
Limitation	n/a	

Scénario : Ecrire des données dans un fichier LDIF

Ce scénario est constitué de deux composants et a pour objectif d'extraire les données d'une table de base de données puis d'écrire ses données dans un fichier LDIF de sortie.

- Cliquez et déposez les composants **tMysqlInput** et **tFileOutputLDIF** de la Palette dans l'espace de modélisation. Connectez les deux composants à l'aide d'un lien **Row > Main**.
- Sélectionnez le composant **tMysqlInput** et cliquez sur l'onglet **Basic settings** de la vue **Component**.
- Si vous conservez les informations de connexion à votre base de données dans la zone **Metadata** du **Repository**, sélectionnez l'option **Repository** dans le champ **Property type** . De même, dans le champ **Schema type**, sélectionnez le schéma correspondant parmi les métadonnées si votre schéma est également stocké dans le **Repository**. Tous les paramètres de connexion comme de schéma sont ainsi récupérés.

- Sinon, sélectionnez **Built-in** dans les champs **Property type** et **Schema** et paramétrez manuellement votre connexion à la base de données et les informations du schéma.
- Puis double-cliquez sur le composant **tFileOutpuLDIF** et paramétrez ses propriétés dans l'onglet **Basic settings**.
- Renseignez le nom et le chemin d'accès au fichier LDIF. S'il n'existe pas, le Job se chargera de le créer.
- Dans le champ **Wrap**, saisissez le nombre de caractères contenus sur une ligne. Les caractères qui suivent seront automatiquement placés sur la ligne suivante.

- Sélectionnez **Add** dans le champ **Change Type** puisque, par définition, un fichier nouvellement créé est vide. Dans le cas d'une modification, vous devrez définir la nature de la modification que vous souhaitez effectuer dans le fichier.
- Dans le champ **Schema**, sélectionnez l'option **Built-in** et cliquez sur **Sync Columns** pour récupérer la définition du schéma d'entrée.
- Appuyez sur **F6** pour exécuter le Job.

Le fichier LDIF créé comporte les données de la table de la base de données et le type de changement effectué sur le fichier est, dans ce cas, défini comme : *addition*.

tFileOutputMSDelimited

Propriétés du tFileOutputMSDelimited

Famille du composant	File/Output	
Fonction	Le composant **tFileOutputMSDelimited** permet d'écrire des schémas multiples dans un fichier délimité.	
Objectif	Le **tFileOutputMSDelimited** crée un fichier délimité multi structure complexe à l'aide de la structure des données (schémas) venant des différents flux d'entrée de type **Row**.	
Basic settings	*File Name*	Chemin d'accès et nom du fichier de sortie, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Row Separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Field Separator	Caractère, chaîne ou expression régulière séparant les champs.
	Use Multi Field Separators	Cochez cette case attribuer un séparateur différent à chacun des schémas, à partir de la colonne **Field separator** du champ **Schemas**.
	Schemas	La table est automatiquement chargée avec des schémas résultant des différents flux d'entrée connectés au composant **tFileOutputMSDelimited**. Renseignez les relations de dépendance entre les différents schémas : **Parent row** : Saisissez le nom du flux père (en fonction du nom du lien Row qui transfère les données). **Parent key column** : Saisissez le nom de la colonne clé du flux père. **Key column** : Saisissez le nom de la colonne clé pour la ligne sélectionnée.
Advanced settings	*Advanced separator (for numbers)*	Cochez cette option pour modifier les séparateurs utilisés pour les nombres : **Thousands separator** : définissez le séparateur utilisé pour les milliers. **Decimal separator** : définissez le séparateur utilisé pour les décimaux.
	CSV options	Cochez cette case pour prendre en compte les paramètres spécifiques aux fichiers CSV, notamment la manière de protéger les caractères dans le champ **Escape char** et le type de guillemet dans le champ **Text enclosure**.
	Create directory if not exists	Cette case est cochée par défaut. Cette option permet de créer le dossier contenant le fichier délimité de sortie s'il n'existe pas déjà.

	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	Don't generate empty file	Cochez cette case pour annuler la génération du fichier si celui-ci est vide.
	tStatCatcher Statistics	Cochez cette case pour collecter les métadonnées de process du Job, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Utilisez ce composant pour écrire un fichier délimité multi structure et séparer les champs à l'aide d'un séparateur de champ.	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tFileOutputMSPositional

Propriétés du tFileOutputMSPositional

Famille du composant	File/Output	
Fonction	Le composant **tFileOutputMSPositional** permet d'écrire des schémas multiples dans un fichier positionnel.	
Objectif	Le **tFileOutputMSPositional** crée un fichier multi structure complexe à l'aide de la structure des données (schémas) venant des différents flux d'entrée de type **Row**.	
Basic settings	*File Name*	Chemin d'accès et nom du fichier à créer, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Row separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Schemas	La table est automatiquement chargée avec des schémas résultant des différents flux d'entrée connectés au composant **tFileOutputMSPositional**. Renseignez les relations de dépendance entre les différents schémas : **Parent row** : Saisissez le nom du flux père (en fonction du nom du lien Row qui transfère les données). **Parent key column** : Saisissez le nom de la colonne clé du flux père. **Key column** : Saisissez le nom de la colonne clé pour la ligne sélectionnée. **Pattern** : Saisissez le modèle (pattern) qui permet de positionner les séparateurs de champ pour chaque ligne d'entrée. **Padding char** : Saisissez entre guillemets le caractère de remplissage à utiliser. Le caractère par défaut est un espace. **Alignment** : Sélectionnez le paramètre d'alignement approprié.
Utilisation	Utilisez ce composant pour écrire un fichier positionnel multi structure et séparer les champs à l'aide d'un séparateur de position.	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tFileOutputMSXML

Propriétés du tFileOutputMSXML

Famille de composant	File/Output	
Fonction	Le composant **tFileOutputMSXML** permet d'écrire des schémas multiples dans un fichier XML structuré.	
Objectif	**tFileOutputMSXML** crée un fichier XML multi structure complexe à l'aide de la structure des données (schémas) venant des différents flux d'entrée de type **Row**.	
Basic settings	*File Name*	Chemin d'accès et nom du fichier à créer, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Configure XML tree	Ouvre l'interface dédiée pour vous aider à mettre en place le mapping XML. Pour plus de détails sur l'interface, voir *Définir un arbre XML Multischéma, page 1001*.
Limitation	n/a	

Définir un arbre XML Multischéma

Double-cliquez sur le composant **tFileOutputMSXML** pour ouvrir l'interface dédiée ou cliquez sur le bouton **[...]** de l'onglet **Basic settings** de la vue **Component**.

La liste déroulante de la zone **Linker source**, à gauche de l'interface, contient tous les schémas d'entrée qui peuvent être ajoutés au fichier XML multischéma de sortie (à la condition que plusieurs flux d'entrée soient connectés au composant **tFileOutputMSXML**).

Dans le champ **Schema List**, on retrouve la liste de toutes les colonnes récupérées du flux d'entrée de la sélection.

Sur la droite de l'interface, définissez la structure XML que vous souhaitez obtenir dans le fichier XML en sortie.

Vous pouvez la créer manuellement ou tout simplement importer la structure XML. Puis importez les colonnes de chaque schéma d'entrée de la sélection dans l'élément de l'arbre XML correspondant.

Importer un arbre XML

Le meilleur moyen de renseigner l'arbre XML est d'importer un fichier XML bien formé.

- Dans la colonne **XML Tree**, cliquez une fois sur le champ **root tag**, qui s'affiche par défaut, pour le renommer.
- Cliquez droit sur ce champ pour afficher le menu contextuel.
- Dans le menu, sélectionnez **Import XML tree**.
- Sélectionnez le fichier à importer et cliquez sur **OK**.

La colonne **XML Tree** est donc automatiquement renseignée avec les bons éléments. Vous pouvez supprimer et ajouter des éléments ou des sous éléments à l'arbre :

- Sélectionnez l'élément adéquat dans l'arbre.
- Cliquez-droit pour faire apparaître le menu contextuel.
- Sélectionnez **Delete** pour supprimer la sélection de l'arbre ou sélectionnez l'option adéquate parmi les suivantes : **Add sub-element**, **Add attribute**, **Add namespace** pour enrichir l'arbre.

Créer manuellement l'arbre XML

Si vous ne possédez pas de structure XML déjà définie, vous pouvez la créer manuellement.

- Dans la colonne **XML Tree**, cliquez une fois sur le champ **root tag**, qui s'affiche par défaut, pour le renommer.
- Cliquez-droit sur ce champ pour afficher le menu contextuel.
- Dans le menu, sélectionnez **Add sub-element** pour créer le premier élément de la structure.

Vous pouvez aussi ajouter un attribut ou un élément enfant à n'importe quel élément de l'arbre ou supprimer n'importe quel élément de l'arbre.

- Sélectionnez l'élément adéquat dans l'arbre que vous venez de créer.
- Cliquez-droit à gauche du nom de l'élément pour afficher le menu contextuel.
- Dans le menu, sélectionnez l'option adéquate parmi les suivantes : **Add sub-element**, **Add attribute**, **Add namespace** ou **Delete**.

Mapping de données XML à partir de sources multischéma

Une fois votre arbre XML créé, sélectionnez le premier schéma à mapper.

Vous pouvez alimenter chaque élément ou sous-élément XML avec les colonnes du flux d'entrée dans la colonne **Related Column** :

- Cliquez sur l'une des entrées de la colonne **Schema column name**.
- Glissez-la dans le sous-élément correspondant à droite.
- Relâchez-la pour que le mapping soit effectif.

Une flèche bleue apparaît pour illustrer ce mapping. S'il est disponible, utilisez le bouton **Auto-Map** situé en bas à gauche de l'interface, il permet d'effectuer cette opération automatiquement.

Vous pouvez déconnecter n'importe quel mapping de n'importe quel élément de l'arbre XML :

- Sélectionnez l'élément de l'arbre XML que vous souhaitez déconnecter de sa source respective.

- Cliquez-droit à gauche de son nom pour afficher le menu contextuel.
- Sélectionnez **Disconnect linker**.

La flèche bleue disparaît.

Définir le statut du noeud

Définir l'arbre XML et le transfert de données ne suffit pas. Vous devez aussi définir les éléments sur lesquels la boucle est effectuée **pour chacune des sources de la sélection** et, si nécessaire, l'élément à partir duquel le regroupement est effectué.

Elément Boucle

L'élément Boucle permet de définir l'élément objet d'une itération. L'élément Boucle est généralement un générateur de lignes.

Pour définir un élément comme un élément Boucle :

- Sélectionnez l'élément adéquat dans l'arbre XML.
- Cliquez-droit à gauche du nom de l'élément pour afficher le menu contextuel.
- Sélectionnez l'option **Set as Loop Element**.

La colonne **Node Status** affiche le nouveau statut sélectionné.

Il ne peut y avoir qu'un seul élément Boucle à la fois.

Elément Group

L'élément Group est optionnel, il représente un élément constant sur lequel est effectuée une fonction Groupby. L'élément Group ne peut être défini que si un élément Boucle a été préalablement défini.

Lorsque vous utilisez un élément Group, les lignes doivent être triées pour pouvoir être regroupées par le noeud sélectionné.

Pour définir un élément comme élément Group :

- Sélectionnnez l'élément adéquat dans l'arbre XML.
- Cliquez-droit à gauche du nom de l'élément pour afficher le menu contextuel.
- Sélectionnez l'option **Set as Group Element**.

La colonne **Node Status** affiche le nouveau statut sélectionné et tout autre statut de regroupement est automatiquement défini, si nécessaire.

Une fois le mapping terminé, cliquez sur **OK** pour valider les paramètres et continuez la configuration de votre Job.

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tFileOutputPositional

Propriétés du tFileOutputPositional

Famille de composant	File/Output	
Fonction	Le composant **tFileOutputPositional** écrit un fichier ligne par ligne en se basant sur la longueur et le format des champ ou colonnes d'une ligne.	
Objectif	Il écrit un fichier ligne par ligne, en se basant sur la structure des données (schéma) venant du flux d'entrée.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier où sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	File Name	Chemin d'accès et nom du fichier de sortie, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma sera créé et conservé pour ce composant. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Row separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Include header	Cochez cette case pour tenir compte de l'en-tête dans le fichier de sortie.
	Compress as zip file	Cochez cette case pour compresser en zip le fichier de sortie.

	Formats	Personnalisez le format des données du fichier positionnel et renseignez les colonnes du tableau **Formats**. **Column** : Sélectionnez la colonne que vous souhaitez personnaliser. **Size** : Saisissez la taille correspondant à la colonne. **Padding char** : Saisissez entre guillemets le caractère de remplissage à utiliser. Le caractère par défaut est un espace. **Alignment** : Sélectionnez le paramètre d'alignement approprié. **Keep** : Si les données contenues dans la colonne ou dans le champ sont trop longues, sélectionnez la partie que vous souhaitez garder.
Advanced settings	*Advanced separator*	Cochez cette case pour modifier les séparateurs de lignes utilisés.
	Create directory if not exists	Cochez cette case pour créer un nouveau dossier si votre dossier de destination n'existe pas.
	Custom the flush buffer size	Cochez cette case pour personnaliser le nombre de lignes à écrire avant de vider le tampon. **Row Number** : Nombre de lignes à écrire avant de vider le tampon.
	Output in row mode	Cochez cette case pour que l'écriture se fasse en mode ligne.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
	Don't generate empty file	Cochez cette case pour empêcher la génération d'un fichier vide.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du Job, ainsi qu'au niveau de chaque composant.
Utilisation	Utilisez ce composant pour lire un fichier et séparer les champs à l'aide du séparateur spécifié.	

Scénario associé :

Pour un scénario associé, consultez *Scénario : Transformer un fichier positionnel en XML, page 958*.

tFileOutputProperties

Propriétés du tFileOutputProperties

Famille de composant	File/Output	
Fonction	Le composant **tFileInputProperties** écrit un fichier de configuration de type ini ou properties.	
Objectif	Il écrit un fichier de configuration contenant des données texte organisées sur le modèle clé=valeur.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Pour ce composant, le schéma est en lecture seule puisque les fichiers sont structurés sur le modèle clé=valeur.
	File format	Sélectionnez le format de votre fichier **.properties** ou **.ini**.
		.properties : fichier de configuration dont les données sont disposées sur deux lignes et structurées de la manière suivante clé=valeur.
		.ini : fichier de configuration dont les données sont disposées sur deux lignes, structurées de la manière suivante clé=valeur. et regroupées en sections.. **Section Name** : saisissez le nom de la section sur laquelle effectuer l'opération d'itération.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	File Name	Chemin d'accès et nom du fichier de sortie, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
Utilisation	Utilisez ce composant pour écrire un fichier dont les données sont organisées selon la structure clé=valeur.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Lire et mapper la clé et les valeurs de fichiers properties et alimenter un glossaire, page 961* du composant *tFileInputProperties*.

tFileOutputXML

Le **tFileOutputXML** appartient à deux familles de composants : File et XML. Pour plus d'informations concernant le composant **tFileOutputXML**, consultez *tFileOutputXML, page 1470*.

tFileProperties

Propriétés du tFileProperties

Famille du composant	File/Management	
Fonction	Le composant **tFileProperties** permet de créer un flux monoligne affichant les propriétés du fichier traité.	
Objectif	Le **tFileProperties** permet de recueillir les informations concernant les propriétés principales d'un fichier défini.	
Basic settings	*Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Edit schema	:Le nombre de lignes en lecture seule est différent sous Java et sous Perl.
	File	Chemin d'accès et nom du fichier à traiter, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Calculate MD5 Hash	Cochez cette case pour vérifier le MD5 du fichier téléchargé.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en standalone.	

Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** Talend Open Studio.
Limitation	n/a	

Scénario : Afficher les propriétés d'un fichier traité

Ce scénario Java décrit un Job très simple permettant d'afficher les propriétés d'un fichier spécifié.

- A partir de la **Palette**, cliquez-déposez les composants **tFileProperties** et **tLogRow** dans l'éditeur graphique.
- Cliquez-droit sur le **tFileProperties** et connectez-le au **tLogRow** à l'aide d'un lien de type **Row Main**.

- Dans l'éditeur graphique, sélectionnez le composant **tFileProperties**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tFileProperties**.

- Configurez le champ **Schema** en mode **Built-In**.

- Cliquez sur le bouton **[...]** du champ **Edit schema** si vous voulez voir les colonnes en lecture seule.
- Dans le champ **File**, saisissez le chemin d'accès ou parcourez vos dossiers jusqu'au fichier dont vous voulez afficher les propriétés.
- Dans l'éditeur graphique, sélectionnez le **tLogRow** et cliquez sur la vue **Component** pour en définir la configuration de base (**Basic settings**). Pour plus d'informations, consultez le composant *tLogRow, page 1163*.
- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

```
Starting job File_Properties at 14:57 26/08/2008.
.-----------------------------------------------.
|                #1. tLogRow_1                  |
+---------------+-------------------------------+
| key           | value                         |
+---------------+-------------------------------+
| abs_path      | C:\test5.txt                  |
| dirname       | C:\                           |
| basename      | test5.txt                     |
| mode_string   | rw                            |
| size          | 86                            |
| mtime         | 1219674736421                 |
| mtime_string  | Mon Aug 25 16:32:16 CEST 2008 |
+---------------+-------------------------------+

Job File_Properties ended at 14:57 26/08/2008. [exit code=0]
```

Les propriétés du fichier défini s'affichent dans la console **Run**.

tFileRowCount

Propriétés du tFileRowCount

Famille de composant	File/Management	
Fonction	Le composant **tFileRowCount** compte le nombre de lignes dans un fichier.	
Objectif	Ce composant ouvre un fichier et le lit ligne par ligne afin de déterminer le nombre de lignes qu'il contient.	
Basic settings	*File Name*	Chemin d'accès et nom du fichier à traiter, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Row separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Ignore empty rows	Cochez cette case pour que les lignes vides ne soient pas prises en compte.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Le **tFileRowCount** est un composant standalone, et doit être utilisé avec un lien de type **OnSubjobOk** vers un **tJava**.	
Global Variables		**Number of counted lines** : retourne le nombre de lignes dans un fichier. Disponible comme variable **Flow**. Retourne un entier. Pour plus d'informations concernant les variables, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main. **Trigger** : OnSubjobOk, OnSubjobError, Run if. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject, Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.

Scénario associé :

Aucun scénario n’est disponible pour ce composant.

tFileTouch

Propriétés du tFileTouch

Famille de composant	File/Management	
Fonction	Le composant **tFileTouch** crée un fichier vide.	
Objectif	Ce composant crée un fichier vide, et crée le répertoire de destination s'il n'existe pas.	
Basic settings	*File Name*	Chemin d'accès et nom du fichier à créer, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Create directory if not exists	Cochez cette case pour créer un nouveau dossier si votre dossier de destination n'existe pas.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en standalone.	
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject, Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.

Scénario associé :

Aucun scénario n'est disponible pour ce composant.

tFileUnarchive

Propriétés du tFileUnarchive

Famille de composant	File/Management	
Fonction	Le composant **tFileUnarchive** dézippe un fichier archive en fonction des paramètres définis et le place dans le répertoire souhaité.	
Objectif	Ce composant dézippe un fichier archive de n'importe quel format (zip, rar, etc.) pour un éventuel traitement.	
Basic settings	*Archive file*	Chemin d'accès au fichier archive.
	Extraction Directory	Répertoire dans lequel le fichier dézippé sera placé.
Java uniquement	*Use archive name as root directory*	Cochez cette case pour créer un dossier du même nom que l'archive, s'il n'existe pas, dans le répertoire spécifié et extraire le(s) fichier(s) dézippé(s) dans ce dossier.
Perl uniquement	*Use Command line tools*	Cochez cette case pour utiliser un autre dézippeur que celui utilisé par défaut par Perl.
Java uniquement	*Check the integrity before unzip*	Cochez cette case pour effectuer une vérification d'intégrité avant de dézipper l'archive.
Java uniquement	*Extract file paths*	Cochez cette case pour reproduire la structure du chemin d'accès au fichier zippé dans l'archive.
Java uniquement	*Need a password*	Cochez cette case et saisissez le mot de passe correct si la décompression de l'archive est protégée par un mot de passe. Notez que l'archive cryptée doit être créée par le composant **tFileArchive**, sinon des messages d'erreur apparaîtront ou rien ne sera extrait, même si aucun message d'erreur n'est apparu.
Utilisation	Ce composant peut être utilisé en standalone mais il peut aussi être utilisé en composant de départ dans un Job via un lien **Iterate**.	
Global Variables		**Current File** : retourne le nom du fichier archive décompressé. Disponible comme variable **Flow**. Retourne une chaîne de caractères. **Current File Path** : retourne le chemin d'accès au fichier archive décompressé. Disponible comme variable **Flow**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Types de connexions* dans le **Guide Utilisateur** Talend Open Studio.

Connections		Liens de sortie (d'un composant à un autre) : **Row** : Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate. **Trigger** : Run if, OnSubjobOk, OnSubjobError, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	n/a	

Scénario associé

Pour un exemple d'utilisation du composant **tFileUnarchive**, consultez *tFileCompare, page 896*.

tGPGDecrypt

Propriétés du tGPGDecrypt

Famille de conposant	File/Management	
Fonction	Le composant **tGPGDecrypt** décrypte un fichier crypté GnuPG et sauvegarde le fichier décrypté dans le répertoire cible spécifié.	
Objectif	Ce composant appelle la commande `gpg -d` afin de décrypter un fichier crypté GnuPG et sauvegarde le fichier décrypté dans le répertoire spécifé.	
Basic settings	*Input encrypted file*	Chemin d'accès au fichier crypté file.
	Output decrypted file	Chemin d'accès au fichier décrypté.
	GPG binary path	Chemin d'accès à la commande GPG.
Perl uniquement	*Secret key*	Saisissez votre clé secrète.
	Passphrase	Saisissez la phrase secrète utilisée pour crypter le fichier d'entrée spécifié.
Java uniquement	*No TTY Terminal*	Cochez cette case pour spécifier qu'aucun terminal TTY n'est utilisé en ajoutant l'option `--no-tty` à la commande de décryptage.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en standalone.	
Limitation	n/a	

Scénario : Décrypter un fichier crypté GnuPG et afficher son contenu

Le scénario suivant décrit un Job de trois composants qui décrypte un fichier crypté GnuPG et affiche son contenu dans la console de la vue **Run**.

- Déposez un composant **tGPGDecrypt**, un **tFileInputDelimited** et un **tLogRow** de la **Palette** dans l'espace de modélisation graphique.
- Reliez le composant **tGPGDecrypt** au **tFileInputDelimited** à l'aide d'un lien **Trigger** > **OnSubjobOk** puis connectez le **tFileInputDelimited** au **tLogRow** en utilisant un lien **Row** > **Main**.
- Double-cliquez sur le composant **tGPGDecrypt** afin d'ouvrir sa vue **Component** et définir ses propriétés :

- Dans le champ **Input encrypted file**, parcourez votre répertoire jusqu'au fichier à décrypter.
- Dans le champ **Output decrypted file**, saisissez le chemin d'accès au fichier décrypté.
- Dans le champ **GPG binary path**, parcourez votre répertoire jusqu'au fichier de commandes GPG.
- Dans le champ **Passphrase**, saisissez la phrase secrète utilisée lors du cryptage du fichier d'entrée.
- Double-cliquez sur le composant **tFileInputDelimited** afin d'ouvrir sa vue **Component** et définir ses propriétés :

- Dans la liste **Property Type**, sélectionnez **Built-In**.
- Dans le champ **File name/Stream**, définissez le chemin d'accès au fichier décrypté, le chemin de sortie défini dans votre composant **tGPGDecrypt**.
- Dans le champ **Header**, définissez le nombre de lignes à ignorer au début du fichier. Dans cet exemple, saisissez *1*.
- Dans le champ **Footer**, saisissez le nombre de lignes à ignorer à la fin du fichier. Dans cet exemple, aucune ligne n'est à ignorer, saisissez donc *0*.
- Dans le champ **Limit**, saisissez le nombre de lignes à traiter. Saisissez *10*.
- Dans la liste **Schema**, sélectionnez **Built-In**. Cela signifie que les propriétés que vous configurez dans le composant ne seront pas réutilisables dans d'autres Jobs.
- Cliquez sur **Edit schema** et éditez le schéma du composant. Cliquez deux fois sur le bouton **[+]** pour ajouter deux colonnes que vous nommerez *idState* et *labelState*.
- Cliquez sur **OK** afin de valider vos modifications et fermer l'éditeur.

- Double-cliquez sur le composant **tLogRow** afin d'afficher sa vue **Component** et définir ses propriétés.

- Dans la liste **Schema**, sélectionnez **Built-In**.
- Dans la zone **Mode**, sélectionnez l'option **Table (print values in cells of a table)**.
- Sauvegardez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job tGPGDecrypt at 18:49
17/11/2010.

[statistics] connecting to socket on port
3990
[statistics] connected
.-------+----------.
|     tLogRow_1     |
|=------+----------=|
|idState|labelState |
|=------+----------=|
|1      |Alabama    |
|2      |Alaska     |
|3      |Arizona    |
|4      |Arkansas   |
|5      |California |
|6      |Colorado   |
|7      |Connecticut|
|8      |Delaware   |
|9      |Florida    |
|10     |Georgia    |
`-------+----------'

[statistics] disconnected
Job tGPGDecrypt ended at 18:49 17/11/2010.
[exit code=0]
```

Le fichier spécifié est décrypté et le nombre défini de lignes s'affiche dans la console de la vue **Run**.

tPivotToColumnsDelimited

Propriétés du tPivotToColumnsDelimited

Famille de composant	File/Output	
Fonction	Le composant **tPivotToColumnsDelimited** extrait les données en fonction d'une opération d'agrégation exercée sur une colonne pivot.	
Objectif	Le **tPivotToColumnsDelimited** permet d'ajuster la sélection de données à transmettre en sortie.	
Basic settings	*Pivot column*	Sélectionnez la colonne qui sert de pivot pour l'opération d'agrégation dans le flux d'entrée.
	Aggregation column	Sélectionnez la colonne contenant les données à agréger dans le flux d'entrée.
	Aggregation function	Sélectionnez la fonction à utiliser dans le cas où plusieurs valeurs sont disponibles dans la colonne pivot.
	Group by	Définissez les ensembles d'agrégation dont les valeurs qui sont utilisées pour les calculs.
		Input Column : Faites le lien entre les libellés des colonnes d'entrée et ceux des colonnes de sortie, dans le cas où vous souhaitez que les libellés du schéma de sortie soient différents du schéma d'entrée.
	File Name	Chemin d'accès et nom du fichier de sortie, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Field separator	Caractère, chaîne ou expression régulière pour séparer les champs du fichier de sortie.
	Row separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
Utilisation	Ce composant nécessite un flux d'entrée.	
Limitation	n/a	

Scénario : Utiliser une colonne pivot pour agréger des données

Le scénario suivant décrit un Job qui agrège des données d'un fichier d'entrée délimité à l'aide d'une colonne pivot.

- Cliquez sur les composants suivant et déposez-les dans l'espace de modélisation : **tFileInputDelimited**, **tPivotToColumnsDelimited**.
- Le fichier à utiliser en entrée est composé de 3 colonnes : *ID*, *Question* et les réponses correspondantes : *Answer*

Id	Question	Answer
1	Name	Juan
2	Name	Jean
3	Name	John
1	Gender	M
2	Gender	F
3	Gender	M
1	Surgery	Yes
2	Surgery	No
3	Surgery	Yes
1	Age	45
2	Age	23
3	Age	42
1	Name	Mary

- Dans l'espace de modélisation, sélectionnez le composant **tFileInputDelimited**.
- Définissez ses propriétés dans l'onglet **Basic settings** de la vue **Component**.

- Sélectionnez le fichier d'entrée pour renseigner le champ **File Name**.
- Définissez les séparateurs de lignes (**Row**) et de champs (**Field**), respectivement : retour chariot et point virgule, dans cet exemple.
- Le fichier contient un en-tête, ainsi définissez-le.
- Paramétrez le schéma avec les trois colonnes : *ID*, *Questions*, *Answers*.

- Puis sélectionnez le composant **tPivotToColumnsDelimited** et paramétrez ses propriétés dans l'onglet **Basic Settings** de la vue **Component**.

- Dans le champ **Pivot column**, sélectionnez la colonne pivot dans le schéma d'entrée. La colonne pivot est généralement celle présentant le plus de doublons (valeurs d'agrégation pivot).
- Dans le champ **Aggregation column**, sélectionnez la colonne à agréger dans le schéma d'entrée.
- Dans le champ **Aggregation function**, sélectionnez la fonction à effectuer lorsque des doublons sont trouvés.
- Dans la table **Group by**, ajoutez une entrée dans la colonne **Input column**, qui sera utilisée pour regrouper la colonne agrégée.
- Dans le champ **File Name**, indiquez le chemin d'accès et le nom du fichier de sortie. Et dans les champs **Row** et **Field separator**, paramétrez les séparateurs pour les lignes et données de sortie ayant été agrégées.

Puis, appuyez sur **F6** pour exécuter le Job. Le fichier de sortie affiche les données qui viennent d'être agrégées.

id	Name	Gender	Surgery	Age
1	Mary	M	Yes	45
2	Jean	M	No	23
3	John	F	Yes	42

Composants Internet

Ce chapitre passe en revue les propriétés des principaux composants présents dans la famille **Internet** de la **Palette** de **Talend Open Studio**.

La famille **Internet** regroupe les composants vous permettant d'accéder à du contenu stocké sur Internet, des services Web aux e-mails en passant par les flux RSS, le SCP, le MOM, le FTP, etc.

tFileFetch

Propriétés du tFileFetch

Famille de composant	Internet	
Fonction	Le composant **tFileFetch** récupère un fichier à partir d'un protocole.	
Objectif	Ce composant permet de récupérer des données d'un fichier à partir d'un protocole.	
Basic settings	*Protocol*	Sélectionnez parmi la liste suivante le protocole à utiliser, puis renseignez les champs correspondants : **http**, **https**, **ftp**, **smb**. Les propriétés peuvent différer légèrement en fonction du type de protocole sélectionné. Les champs additionnels sont définis dans ce tableau à la suite des propriétés générales.
	URI	Saisissez l'URI du site sur lequel le fichier doit être récupéré.
	Use cache to save the resource	Cochez cette case pour enregistrer les données dans le cache. Cette option vous permet de traiter directement le flux de données du fichier (en mode stream) sans sauvegarder ce fichier sur votre disque. Elle permet donc un gain de temps et de performance.
smb	*Domain*	Saisissez le nom du domaine du serveur Microsoft.
smb	*Username* et *Password*	Saisissez l'identifiant utilisateur et le mot de passe de connexion au serveur.
	Destination directory	Sélectionnez le répertoire de destination dans lequel le fichier récupéré sera placé.
	Destination Filename	Saisissez le nouveau nom du fichier récupéré.
http, https, ftp	*Create full path according to URI*	Cette case est cochée par défaut elle permet de reproduire l'arborescence de répertoires de l'URI. Pour enregistrer le fichier à la racine de votre répertoire de destination, décochez la case.
http, https	*Add header*	Cochez cette case si vous souhaitez ajouter un ou plusieurs en-tête(s) comme conditions de récupération. Dans le tableau **Headers**, saisissez le(s) paramètre(s) d'en-tête HTTP dans le champ **Name** et la valeur correspondante dans le champ **Value**.

http, https	*POST method*	Cette case est cochée par défaut, elle permet d'utiliser la méthode POST. Dans le tableau **Parameters**, saisissez le nom de la ou des variable(s) dans le champ **Name** et leur valeur correspondante dans le champ **Value**. Décochez cette case si vous souhaitez utiliser la méthode GET.
http, https, ftp	*Die on error*	Décochez cette case pour passer les lignes en erreur et terminer le traitement des lignes sans erreur.
http, https, ftp	*Read Cookie*	Cochez cette case pour que le **tFileFetch** charge un cookie d'authentification Web.
http, https, ftp	*Save Cookie*	Cochez cette case pour sauvegarder le cookie d'authentification Web. Cela signifie que vous n'aurez plus à vous identifier sur le même site Web.
http, https, ftp	*Cookie directory*	Cliquez sur le bouton **[...]** et parcourez votre répertoire jusqu'à l'endroit où vous souhaitez sauvegarder votre cookie, ou jusqu'à l'endroit où il est déjà sauvegardé.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
http, https	*Timeout*	Nombre de secondes avant fermeture de la connexion au protocole.
http, https	*Print response to console*	Cochez cette case pour imprimer la réponse du serveur dans la console.
http, https	*Upload file*	Cochez cette case pour charger un ou plusieurs fichier(s) sur le serveur. Dans le champ **Name**, saisissez le nom du fichier que vous souhaitez charger, et dans le champ **File**, saisissez le chemin d'accès au fichier.
http, https, ftp	*Enable proxy server*	Cochez cette case si vous vous connectez derrière un proxy et renseignez les informations correspondantes.
http, https, ftp	*Enable NTLM Credentials*	Cochez cette case si vous utiliser un protocole d'authentification NTLM. **Domain** : Nom de domaine du client. **Host** : Adresse IP du client.
http, https	*Need authentication*	Cochez la case d'authentification et renseignez le nom de l'utilisateur dans le champs **Username** et son mot de passe dans le champs **Password**, si cela est nécessaire pour accéder au protocole.
http, https	*Support redirection*	Cochez cette case pour répéter la requête de redirection jusqu'à ce que la redirection soit effective et que le fichier puisse être récupéré.
Utilisation	Ce composant est généralement utilisé en composant de départ pour alimenter le flux d'entrée d'un Job et est souvent utilisé avec un lien de type **OnSubjobOk** ou **OnComponentOk**, en fonction du contexte.	
Limitation	n/a	

Scénario 1 : Récupérer des données à partir d'un protocole HTTP

Ce scénario décrit un Job comprenant trois composants, qui retrouve des données à partir d'un site internet HTTP et sélectionne des données qui seront stockées dans un fichier délimité.

- Cliquez et déposez les composants **tFileFetch**, **tFileInputRegex** et **tFileOutputDelimited** dans l'espace de modélisation, puis connectez-les comme illustré sur la figure ci-dessus.
- Dans l'onglet **Basic settings** du composant **tFileFetch**, sélectionnez le protocole que vous souhaitez utiliser. Ici, utilisez le protocole HTTP.
- Saisissez l'URI sur lequel le fichier peut être récupéré.
- Dans le champ **Destination directory**, sélectionnez le répertoire dans lequel le fichier récupéré doit être stocké.
- Dans le champ **Filename**, saisissez le nouveau nom du fichier si vous souhaitez le renommer. Dans cet exemple, *filefetch.txt*.
- Si nécessaire, cochez la case **Add header** et définissez une ou plusieurs requêtes HTTP comme conditions de récupération. Par exemple, vous pouvez récupérer le fichier s'il a été modifié depuis une date précise.
  Dans le tableau **Headers**, dans le champ **Name**, saisissez *"If-Modified-Since"* puis saisissez *"Sat, 29 Oct 1994 19:43:31 GMT"* dans le champ **Value**.
  Pour plus d'informations concernant la définition d'en-têtes HTTP, consultez *Header Field Definitions* (en anglais).
- Sélectionnez le composant **tFileInputRegex**. Dans le champ **File name** paramétrez le nom du fichier afin qu'il corresponde au fichier récupéré précédemment.
- Dans le champ **Regex**, utilisez une expression régulière pour sélectionner les données adéquates du fichier récupéré. Dans cet exemple :
  ```
  <td(?: class="leftalign")?> \s* (t\w+) \s* </td>
  ```

Utilisez la bonne expression régulière en fonction du langage de génération utilisé. En effet, la syntaxe est différente en Java et en Perl, ainsi que les guillemets utilisées pour introduire les expressions régulières : guillemets simples en Perl et guillemets doubles en Java.

- Si nécessaire, définissez l'en-tête dans le champ **Header**, le pied de page dans le champ **Footer** et le nombre de lignes à traiter dans le champ **Limit**. Pour cet exemple, ignorez ces champs.

- Définissez le schéma décrivant le flux à transmettre au composant de sortie.
- Le schéma devrait être automatiquement propagé au flux de sortie, mais pour en être sûr, vérifiez le schéma dans l'onglet **Basic settings** du composant **tFileOutputDelimited**.
- Puis appuyez sur **F6** pour exécuter le Job.

Scénario 2 : Réutiliser un cookie stocké pour récupérerdes fichiers via un protocole HTTP

Ce scénario décrit un Job comprenant deux composants, qui se connecte à un site Web HTTP donné et, à l'aide d'un cookie stocké dans un répertoire local spécifié, récupère les données de ce site.

- Déposez deux composants **tFileFetch** de la **Palette** dans l'espace de modélisation graphique.
- Reliez ces deux composants à l'aide d'un lien **Trigger** > **OnSubjobOk**
- Double-cliquez sur le **tFileFetch_1** afin d'ouvrir sa vue **Component** et définir ses propriétés.

- Dans la liste **Procotol**, sélectionnez le protocole que vous souhaitez utiliser. Ici, utilisez le protocole **http**.
- Dans le champ **URI**, saisissez l'URI via laquelle vous pouvez vous connecter au site Web et récupérer la page Web. Dans cet exemple, l'URI est `http://www.codeproject.com/script/Membership/LogOn.aspx?rp=http%3a%2f%2fwww.codeproject.com%2fKB%2fcross-platform%2fjavacsharp.aspx&download=true`.
- Dans le champ **Destination directory**, parcourez votre répertoire jusqu'au dossier dans lequel stocker le fichier récupéré. Ce dossier sera créé à la volée s'il n'existe pas. Dans cet exemple, saisissez *C:/Logpage*.
- Dans le champ **Destination Filename**, saisissez un nouveau nom pour le fichier si vous souhaitez le modifier. Dans cet exemple, saisissez *webpage.html*.
- Sous le tableau **Parameters**, cliquez deux fois sur le bouton **[+]** pour ajouter deux lignes.
- Dans la colonne **Name** du tableau **Parameters**, saisissez respectivement le nom des deux lignes, *Email* et *Password*, requises par le site Web auquel vous vous connectez.
- Dans la colonne **Value**, saisissez les informations correspondantes, respectivement votre adresse e-mail et votre mot de passe.
- Cochez la case **Save cookie** pour activer le champ **Cookie directory**.

- Dans le champ **Cookie directory**, parcourez votre répertoire jusqu'au dossier dans lequel vous souhaitez stocker le cookie puis saisissez un nom pour le cookie à sauvegarder. Le dossier doit déjà exister.
  Dans cet exemple, le répertoire est *C:/temp/Cookie*.
- Double-cliquez sur le **tFileFetch_2** afin d'ouvrir sa vue **Component** et définir ses propriétés.

- Dans la liste **Procotol**, sélectionnez **http**.
- Dans le champ URI, saisissez l'adresse à partir de laquelle récupérer les fichiers qui vous intéressent. Dans cet exemple, l'adresse est `http://www.codeproject.com/KB/java/RemoteShell/RemoteShell.zip`.
- Dans le champ **Destination directory**, parcourez votre répertoire jusqu'au dossier dans lequel vous souhaitez stocker les fichiers récupérés. Ce dossier peut être automatiquement créé s'il n'existe pas durant le processus d'exécution. Dans cet exemple, saisissez *C:/JavaProject*.
- Dans le champ **Destination Filename**, saisissez un nouveau nom pour le fichier si vous souhaitez le modifier. Dans cet exemple, saisissez *RemoteShell.zip*.
- Décochez la case **Post method** pour désactiver le tableau **Parameters**.
- Cochez la case **Read cookie** afin d'activer le champ **Cookie directory**.
- Dans le champ **Cookie directory**, parcourez votre répertoire jusqu'au cookie que vous avez sauvegardé et que vous souhaitez utiliser. Dans cet exemple, le répertoire est *C:/temp/Cookie*.
- Sauvegardez votre Job et appuyez sur **F6** pour l'exécuter.
- Vérifiez les dossiers dans lesquels vous avez stocké les fichiers récupérés.

Scénario associé

Pour un exemple de transfert de données en mode stream, consultez le *Scénario 2 : Lire les données d'un fichier distant en mode stream, page 919*.

tFileInputJSON

Le composant **tFileInputJSON** appartient à deux familles différentes : Internet et File. Pour plus d'informations, consultez *tFileInputJSON, page 935*.

tFTPConnection

Propriétés du tFTPConnection

Famille de composant	Internet/FTP	
Fonction	Le composant **tFTPConnection** ouvre une connexion FTP afin d'effectuer une transaction.	
Objectif	Le **tFTPConnection** permet d'ouvrir une connexion FTP afin d'effectuer des transferts de fichiers en une seule transaction.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Adresse IP du serveur FTP.
	Port	Numéro du port d'écoute du serveur FTP.
	Username	Identifiant de connexion au serveur FTP.
	Password	Mot de passe de connexion au serveur FTP.
	SFTP Support/Anthentication method	Cochez cette case puis, à partir de la liste déroulante **Authentication method**, sélectionnez l'option adéquate d'authentification au serveur : **Password** : Renseignez le mot de passe. **Public key** : Renseignez la clé privée ou cliquez sur le bouton **[...]** à côté du champ **Private key** pour parcourir vos dossiers jusqu'au fichier adéquat. Si vous choisissez l'option **Public Key** comme méthode d'authentification, assurez-vous que la clé ait été ajoutée à l'agent ou qu'aucune passphrase (phrase secrète) ne soit demandée. Lorsque la case est décochée, sélectionnez votre mode de connexion, **Passive** : vous déterminez le port de connexion à utiliser pour permettre le transfert des données, ou **Active** : le serveur FTP détermine le port de connexion à utiliser pour permettre le transfert des données.
Advanced settings	*Use Socks Proxy*	Cochez cette case si vous souhaitez utiliser un proxy, puis paramétrez les champs **Host**, **Port**, **User** et **Password** du proxy.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme sous-Job a un seul composant.	

Limitation	n/a

Scénarios associés

Pour un scénario associé, consultez le *Scénario : Copier des fichiers sur un serveur FTP distant, page 1053*.
Consultez également le *Scénario : Itération sur un répertoire distant, page 1043* du **tFTPFileList**

Pour un scénario associé utilisant un protocole différent, consultez le *Scénario : Récupérer des fichiers d'un serveur SCP distant, page 1098*.

tFTPDelete

Propriétés du tFTPDelete

Famille de composant	Internet/FTP	
Fonction	Le composant tFTPDelete supprime les fichiers sélectionnés via une connexion FTP.	
Objectif	Le **tFTPDelete** supprime les fichiers d'un serveur FTP distant.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection/Component List	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vous pouvez utiliser l'onglet **Dynamic settings**, pour partager la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct à travers les deux niveaux de Job. Pour plus d'informations concernant les **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur FTP.
	Port	Numéro du port d'écoute du serveur FTP.
	Username et *Password*	Informations d'authentification de l'utilisateur du serveur FTP.
	Remote directory	Répertoire source où se trouvent les fichiers à supprimer.

	SFTP Support/Anthentication method	Cochez cette case puis, à partir de la liste déroulante **Authentication method**, sélectionnez l'option adéquate d'authentification au serveur : **Password** : Renseignez le mot de passe. **Public key** : Renseignez la clé privée ou cliquez sur le bouton **[...]** à côté du champ **Private key** pour parcourir vos dossiers jusqu'au fichier adéquat. Si vous choisissez l'option Public Key comme méthode d'authentification, assurez-vous que la clé ait été ajoutée à l'agent ou qu'aucune passphrase (phrase secrète) ne soit demandée. Lorsque la case est décochée, sélectionnez votre mode de connexion, **Passive** : vous déterminez le port de connexion à utiliser pour permettre le transfert des données, ou **Active** : le serveur FTP détermine le port de connexion à utiliser pour permettre le transfert des données.
	Files	Noms ou chemins d'accès aux fichiers à supprimer.
Advanced settings	*Use Socks Proxy*	Cochez cette case si vous souhaitez utiliser un proxy, puis paramétrez les champs **Host**, **Port**, **User** et **Password** du proxy.
	Ignore Failure At Quit (FTP)	Cochez cette case afin d'ignorer les erreurs dues à la fermeture du FTP ou de la bibliothèque.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme sous-Job à un seul composant mais il peut aussi être utilisé comme composant de sortie ou de fin.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé, consultez le *Scénario : Copier des fichiers sur un serveur FTP distant, page 1053*.

Pour un scénario associé utilisant un protocole différent, consultez le *Scénario : Récupérer des fichiers d'un serveur SCP distant, page 1098*.

tFTPFileExist

Propriétés du tFTPFileExist

Famille de composant	Internet/FTP	
Fonction	Le composant **tFTPFileExist** vérifie si un fichier est présent sur le serveur FTP.	
Objectif	Le **tFTPFileExist** permet de vérifier la présence d'un fichier sur le serveur FTP distant.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection/Component List	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vous pouvez utiliser l'onglet **Dynamic settings**, pour partager la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct à travers les deux niveaux de Job. Pour plus d'informations concernant les **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur FTP.
	Port	Numéro du port d'écoute du serveur FTP.
	Username et *Password*	Informations d'authentification de l'utilisateur du serveur FTP.
	Remote directory	Répertoire source où se trouvent les fichiers à supprimer.
	File Name	Nom du fichier dont vous voulez vérifier la présence.

	SFTP Support/Anthentication method	Cochez cette case puis, à partir de la liste déroulante **Authentication method**, sélectionnez l'option adéquate d'authentification au serveur : **Password** : Renseignez le mot de passe. **Public key** : Renseignez la clé privée ou cliquez sur le bouton **[...]** à côté du champ **Private key** pour parcourir vos dossiers jusqu'au fichier adéquat. Si vous choisissez l'option Public Key comme méthode d'authentification, assurez-vous que la clé ait été ajoutée à l'agent ou qu'aucune passphrase (phrase secrète) ne soit demandée. Lorsque la case est décochée, sélectionnez votre mode de connexion, **Passive** : vous déterminez le port de connexion à utiliser pour permettre le transfert des données, ou **Active** : le serveur FTP détermine le port de connexion à utiliser pour permettre le transfert des données.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
Advanced settings	*Use Socks Proxy*	Cochez cette case si vous souhaitez utiliser un proxy, puis paramétrez les champs **Host**, **Port**, **User** et **Password** du proxy.
	Ignore Failure At Quit (FTP)	Cochez cette case afin d'ignorer les erreurs dues à la fermeture du FTP ou de la bibliothèque.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme sous-Job à un seul composant mais il peut aussi être utilisé comme composant de sortie ou de fin.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé, consultez le *Scénario : Copier des fichiers sur un serveur FTP distant, page 1053*.

Pour un scénario associé utilisant un protocole différent, consultez le *Scénario : Récupérer des fichiers d'un serveur SCP distant, page 1098*.

tFTPFileList

Propriétés du tFTPFileList

Famille de composant	Internet/FTP	
Fonction	Le composant **tFTPFileList** fait une boucle sur les fichiers et/ou les dossiers d'un répertoire distant donné.	
Objectif	Le **tFTPFileList** récupère un groupe de fichiers et/ou de dossiers suivant un masque donné et fait une boucle sur chacun d'eux, en se connectant à un répertoire distant via un protocole FTP.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection/Component List	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vous pouvez utiliser l'onglet **Dynamic settings**, pour partager la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct à travers les deux niveaux de Job. Pour plus d'informations concernant les **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP.
	Port	Numéro du port d'écoute.
	Username et *Password*	Informations d'authentification de l'utilisateur.
	Remote directory	Chemin d'accès au répertoire distant.
	File detail	Cochez cette case pour voir toutes les informations du fichier, et non pas juste son nom.

	SFTP Support/Anthentication method	Cochez cette case puis, à partir de la liste déroulante **Authentication method**, sélectionnez l'option adéquate d'authentification au serveur : **Password** : Renseignez le mot de passe. **Public key** : Renseignez la clé privée ou cliquez sur le bouton **[...]** à côté du champ **Private key** pour parcourir vos dossiers jusqu'au fichier adéquat. Si vous choisissez l'option **Public Key** comme méthode d'authentification, assurez-vous que la clé ait été ajoutée à l'agent ou qu'aucune passphrase (phrase secrète) ne soit demandée.
	Connect Mode	Sélectionnez le mode adéquat de connexion au serveur : **Active** : vous déterminez le port de connexion à utiliser pour permettre le transfert des données. **Passive** : le serveur FTP détermine le port de connexion à utiliser pour permettre le transfert des données.
Advanced settings	*Use Socks Proxy*	Cochez cette case si vous souhaitez utiliser un proxy, puis paramétrez les champs **Host**, **Port**, **User** et **Password** du proxy.
	Ignore Failure At Quit (FTP)	Cochez cette case afin d'ignorer les erreurs dues à la fermeture du FTP ou de la bibliothèque.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Le composant **tFTPFilelist** est généralement utilisé comme composant seul dans un sous-job mais il peut aussi être utilisé avec d'autres composants.	

Scénario : Itération sur un répertoire distant

Le scénario suivant décrit un Job de trois composants qui a pour objectif de se connecter à un serveur FTP, de lister les fichiers d'un répertoire distant donné en fonction d'un masque, de les récupérer et de les envoyer vers un répertoire local défini.

- Cliquez et déposez les composants suivants : **tFTPConnection**, **tFTPFileList** et **tFTPGet** dans l'espace de modélisation.
- Cliquez droit sur le composant **tFTPConnection** et connectez-le au composant **tFTPFileList** via une connexion de type **OnSubjobOk**. Puis connectez le **tFTPFileList** au **tFTPGet** à l'aide d'une connexion de type **Iterate**.

- Sélectionnez le **tFTPConnection** et cliquez sur l'onglet **Basic settings** de la vue **Component** pour paramétrer le composant :

- Dans le champ **Host**, renseignez l'adresse IP du serveur FTP.
- Dans le champ **Port**, renseignez le numéro du port d'écoute.
- Dans les champs **Username** et **Password**, renseignez respectivement le login et le mot de passe de connexion au serveur distant.
- Dans la liste **Connect Mode**, sélectionnez le mode de connexion au serveur FTP que vous souhaitez utiliser.
- Sélectionnez le **tFTPFileList** et cliquez sur l'onglet **Basic settings** de la vue **Component** pour paramétrer le composant :

- Cochez la case **Use an existing connection** et dans la liste **Component list**, sélectionnez le composant de connexion **tFTPConnection_1**. Les informations de connexion sont automatiquement renseignées.
- Dans le champ **Remote directory**, renseignez le chemin relatif du répertoire contenant les fichiers à lister.

- Dans le champ **Filemask**, indiquez un masque de fichier. Utilisez les caractères spéciaux si besoin est. Dans cet exemple, on ne veut récupérer que les fichiers délimités (*.csv*).
- Dans la liste **Connect Mode**, sélectionnez le mode de connexion au serveur FTP que vous souhaitez utiliser. Dans cet exemple, il s'agit du mode **Passive**.
- Sélectionnez le **tFTPGet** et cliquez sur l'onglet **Basic settings** de la vue **Component** pour paramétrer le composant :

- Cochez la case **Use an existing connection** et dans la liste **Component list**, sélectionnez le composant de connexion adéquat, **tFTPConnection_1** dans cet exemple. Les informations de connexion sont automatiquement renseignées.
- Dans le champ **Local directory**, renseignez le chemin relatif du répertoire de destination des fichiers à récupérer.
- Dans le champ **Remote directory**, renseignez le chemin relatif du répertoire contenant les fichiers à récupérer.
- Dans la liste **Transfer Mode**, sélectionnez le mode de transfert FTP que vous souhaitez utiliser.
- Dans la liste **Overwrite file**, sélectionnez l'une des options disponibles pour le transfert du fichier.
- Dans le champ **Filemask**, sélectionnez la variable globale `((String)globalMap.get("tFTPFileList_1_CURRENT_FILEPATH"))` de manière à traiter tous les fichiers du répertoire source.
- Dans la liste **Connect Mode**, sélectionnez le mode de connexion au serveur FTP que vous souhaitez utiliser.
- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

Une fois la connexion au serveur FTP établie, les fichiers sont listés dans le répertoire spécifié, selon le masque choisi, puis sont récupérés et stockés dans le répertoire de destination préalablement défini. Vous pouvez vérifier que la copie des fichiers a été réalisée à partir de votre explorateur de disque.

tFTPFileProperties

Propriétés du tFTPFileProperties

Famille du composant	Internet	
Fonction	Le composant **tFTPFileProperties** permet de créer un flux monoligne affichant les propriétés du fichier traité sur un serveur FTP.	
Objectif	Le **tFTPFileProperties** permet de recueillir les informations concernant les propriétés principales d'un fichier défini sur un serveur FTP.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Schema type	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Edit schema	:Le nombre de lignes en lecture seule est différent sous Java et sous Perl.
	Use an existing connection/Component List	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies.
	Host	Adresse IP du serveur FTP.
	Port	Numéro du port d'écoute du serveur FTP.
	Username	Login de connexion au serveur FTP.
	Password	Mot de passe de connexion au serveur FTP.
	Local directory	Chemin d'accès à l'emplacement de destination du fichier.
	Remote directory	Chemin d'accès du répertoire source où les fichiers sont récupérés.

	File	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**.
	SFTP Support/Anthenticat ion method	Cochez cette case puis, à partir de la liste déroulante **Authentication method**, sélectionnez l'option adéquate d'authentification au serveur : **Password** : Renseignez le mot de passe. **Public key** : Renseignez la clé privée ou cliquez sur le bouton **[...]** à côté du champ **Private key** pour parcourir vos dossiers jusqu'au fichier adéquat. Si vous choisissez l'option **Public Key** comme méthode d'authentification, assurez-vous que la clé ait été ajoutée à l'agent ou qu'aucune passphrase (phrase secrète) ne soit demandée. Lorsque la case est décochée, sélectionnez votre mode de connexion, **Passive** ou **Active**.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
	Calculate MD5 Hash	Cochez cette case pour vérifier le MD5 du fichier téléchargé.
Advanced settings	*Use Socks Proxy*	Cochez cette case si vous souhaitez utiliser un proxy, puis paramétrez les champs **Host**, **Port**, **User** et **Password** du proxy.
	Ignore Failure At Quit (FTP)	Cochez cette case afin d'ignorer les erreurs dues à la fermeture du FTP ou de la bibliothèque.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en standalone.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Afficher les propriétés d'un fichier traité, page 1012*.

tFTPGet

Propriétés du tFTPGet

Famille de composant	Internet/FTP	
Fonction	Le composant **tFTPGet** récupère les fichiers sélectionnés via une connexion FTP.	
Objectif	Le **tFTPGet** récupère les fichiers sélectionnés d'un serveur FTP distant et les copie dans un répertoire local.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection/Component List	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies.
	Host	Adresse IP du serveur FTP.
	Port	Numéro du port d'écoute du serveur FTP.
	Username	Login de connexion au serveur FTP.
	Password	Mot de passe de connexion au serveur FTP.
	Local directory	Chemin d'accès à l'emplacement de destination du fichier.
	Remote directory	Chemin d'accès du répertoire source où les fichiers sont récupérés.
	Transfer mode	Différents modes de transfert FTP.
	Overwrite file	Liste des options disponibles pour le transfert du fichier. **Append** : Cochez cette case pour écrire les données à la fin de l'enregistrement, et ainsi ne pas l'écraser.

	SFTP Support	Lorsque vous cochez cette case, les champs **Overwrite file** et **Authentication method** s'affichent. **Overwrite file** vous propose trois options : **Overwrite** : pour écraser le fichier déjà existant. **Resume** : pour reprendre le téléchargement du fichier là où il a été interrompu. **Append :** pour écrire à la suite du fichier sans l'écraser. **Authentication** vous propose deux modes d'authentification : **Public key** : Renseignez le chemin d'accès à la clé publique. **Password** : Renseignez le mot de passe.
	FTPS Support	Cochez cette case si vous souhaitez vous connecter au serveur FTP via une connexion FTPS. Deux champs apparaissent : **Keystore File** : Saisissez le chemin d'accès au keystore (fichier protégé par un mot de passe, qui peut contenir plusieurs clés et certificats). **Keystore Password** : saisissez votre mot de passe correspondant au keystore.
	Files	Noms ou chemins d'accès aux fichiers à transférer.
	Connection Mode	Sélectionnez votre mode de connexion. **Passive** : vous déterminez le port de connexion à utiliser pour permettre le transfert des données. **Active** : le serveur FTP détermine le port de connexion à utiliser pour permettre le transfert des données.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et les lignes contenant les erreurs seront ignorées.
Advanced settings	*Use Socks Proxy*	Cochez cette case si vous souhaitez utiliser un proxy, puis paramétrez les champs **Host**, **Port**, **User** et **Password** du proxy.
	Ignore Failure At Quit (FTP)	Cochez cette case afin d'ignorer les erreurs dues à la fermeture du FTP ou de la bibliothèque.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme sous-Job a un seul composant mais il peut aussi être utilisé comme composant de sortie ou de fin.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Copier des fichiers sur un serveur FTP distant, page 1053*.
Consultez également le *Scénario : Itération sur un répertoire distant, page 1043*. du **tFTPFileList**.

Pour un scénario associé utilisant un protocole différent, consultez le *Scénario : Récupérer des fichiers d'un serveur SCP distant, page 1098*.

tFTPPut

Propriétés du tFTPPut

Famille de composant	Internet/FTP	
Fonction	Le composant **tFTPPut** copie les fichiers sélectionnés via une connexion FTP.	
Objectif	Le **tFTPPut** copie les fichiers sélectionnés d'un répertoire local défini vers un répertoire FTP distant.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection/Component List	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vous pouvez utiliser l'onglet **Dynamic settings**, pour partager la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct à travers les deux niveaux de Job. Pour plus d'informations concernant les **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur FTP.
	Port	Numéro du port d'écoute du serveur FTP.
	Username	Login de connexion au serveur FTP.
	Password	Mot de passe de connexion au serveur FTP.
	Local directory	Chemin d'accès au répertoire source du ou des fichier(s).
	Remote directory	Chemin d'accès au répertoire de destination du ou des fichier(s).
	Transfer mode	Différents modes de transfert FTP.
	Overwrite file	Liste des options disponibles pour le fichier transféré. **Append** : Cochez cette case pour écrire les données à la fin de l'enregistrement, et ainsi ne pas l'écraser.

	SFTP Support et *Authentication method*	**SFTP Support**: cochez cette case pour établir une méthode d'authentification. **Authentication method***: sélectionnez la méthode d'authentification à partir de la liste déroulante.*
	Files	Cliquez sur le bouton **[+]** pour ajouter une ligne puis remplissez les colonnes. **Filemask**: noms et chemins d'accès aux fichiers à transférer. **New name**: nom à donner au fichier FTP après le transfert.
	Connection Mode	Sélectionnez votre mode de connexion. **Passive** : vous déterminez le port de connexion à utiliser pour permettre le transfert des données. **Active** : le serveur FTP détermine le port de connexion à utiliser pour permettre le transfert des données.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et les lignes contenant les erreurs seront ignorées.
Advanced settings	*Use Socks Proxy*	Cochez cette case si vous souhaitez utiliser un proxy, puis paramétrez les champs **Host**, **Port**, **User** et **Password** du proxy.
	Ignore Failure At Quit (FTP)	Cochez cette case afin d'ignorer les erreurs dues à la fermeture du FTP ou de la bibliothèque.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme sous-Job a un seul composant mais il peut aussi être utilisé comme composant de sortie ou de fin.	
Limitation	n/a	

Scénario : Copier des fichiers sur un serveur FTP distant

Ce scénario crée un Job à deux composants permettant d'ouvrir une connexion à un serveur distant afin de copier les fichiers souhaités en une seule transaction.

- Cliquez et déposez les composants **tFTPConnection** permettant d'effectuer toutes les opérations en une seule transaction et **tFTPPut** de la famille **Internet** dans le Job Designer.
- Reliez le **tFTPConnection** au **tFTPPut** à l'aide d'un lien de type **OnSubJobOk**.
- Dans le Job designer, double-cliquez sur le **tFTPConnection** pour paramétrer ses propriétés de base dans l'onglet **Basic settings** de la vue **Component** :

- Dans le champ **Host**, renseignez l'adresse IP du serveur.
- Dans le champ **Port**, renseignez le numéro du port d'écoute.
- Dans les champs **Username** et **Password**, renseignez respectivement le login et le mot de passe de connexion au serveur distant.
- Dans la liste **Connection Mode**, sélectionnez le mode de connexion au serveur FTP que vous souhaitez utiliser.
- Dans le Job designer, double-cliquez sur le **tFTPPut** pour paramétrer ses propriétés de base dans l'onglet **Basic settings** de la vue **Component** :

- Cochez la case **Use an existing connection** et dans la liste **Component list**, sélectionnez le composant de connexion **tFTPConnection_1**. Les informations de connexion sont automatiquement renseignées.
- Dans le champ **Local directory**, renseignez le chemin d'accès au répertoire local contenant le fichier, à moins que vous ne renseigniez directement les masques des fichiers dans la colonne **Filemask** du champ **Files**.
- Dans le champ **Remote directory**, renseignez les informations de connexion au répertoire du seveur distant.
- Dans la liste **Transfert mode**, sélectionnez le mode de transfert utilisé.
- Dans la liste **Overwrite file**, sélectionnez l'option d'écrasement du fichier.
- Dans le tableau **Files**, cliquez deux fois sur le bouton **[+]** pour ajouter deux lignes à la liste **Filemask**, renseignez les masques de fichier pour chaque fichier que vous souhaitez copier dans le répertoire distant.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

Les fichiers spécifiés dans le champ **Filemask** sont copiés dans le répertoire du serveur distant.

tFTPRename

Propriétés du tFTPRename

Famille de composant	Internet/FTP	
Fonction	Le composant **tFTPRename** renomme les fichiers sélectionnés via une connexion FTP.	
Objectif	Le **tFTPRename** renomme les fichiers sélectionnés d'un répertoire local défini vers un répertoire FTP distant.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection/Component List	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vous pouvez utiliser l'onglet **Dynamic settings**, pour partager la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct à travers les deux niveaux de Job. Pour plus d'informations concernant les **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur FTP.
	Port	Numéro du port d'écoute du serveur FTP.
	Username	Login de connexion au serveur FTP.
	Password	Mot de passe de connexion au serveur FTP.
	Remote directory	Chemin d'accès au répertoire de destination du ou des fichier(s).
	Overwrite file	Liste des options disponibles pour le fichier transféré. **Append** : Cochez cette case pour écrire les données à la fin de l'enregistrement, et ainsi ne pas l'écraser.

	SFTP Support et *Authentication method*	**SFTP Support** : cochez cette case pour établir une méthode d'authentification. **Authentication method** : sélectionnez la méthode d'authentification à partir de la liste déroulante.
	Files	Cliquez sur le bouton **[+]** pour ajouter une ligne puis remplissez les colonnes. **Filemask** : noms et chemins d'accès aux fichiers à transférer. **New name** : nom à donner au fichier FTP après le transfert.
	Connection Mode	Sélectionnez votre mode de connexion. **Passive** : vous déterminez le port de connexion à utiliser pour permettre le transfert des données. **Active** : le serveur FTP détermine le port de connexion à utiliser pour permettre le transfert des données.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et les lignes contenant les erreurs seront ignorées.
Advanced settings	*Use Socks Proxy*	Cochez cette case si vous souhaitez utiliser un proxy, puis paramétrez les champs **Host**, **Port**, **User** et **Password** du proxy.
	Ignore Failure At Quit (FTP)	Cochez cette case afin d'ignorer les erreurs dues à la fermeture du FTP ou de la bibliothèque.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme sous-Job a un seul composant mais il peut aussi être utilisé comme composant de sortie ou de fin.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Copier des fichiers sur un serveur FTP distant, page 1053* .

tFTPTruncate

Propriétés du tFTPTruncate

Famille de composant	Internet/FTP	
Fonction	Le composant **tFTPTruncate** tronque les fichiers sélectionnés via une connexion FTP.	
Objectif	Le **tFTPTruncate** tronque les fichiers sélectionnés d'un répertoire local défini vers un répertoire FTP distant.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Use an existing connection/Component List	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vous pouvez utiliser l'onglet **Dynamic settings**, pour partager la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct à travers les deux niveaux de Job. Pour plus d'informations concernant les **Dynamic settings**, consultez le **Guide Utilisateur** de votre Studio.
	Host	Adresse IP du serveur FTP.
	Port	Numéro du port d'écoute du serveur FTP.
	Username	Login de connexion au serveur FTP.
	Password	Mot de passe de connexion au serveur FTP.
	Remote directory	Chemin d'accès au répertoire de destination du ou des fichier(s).
	SFTP Support et *Authentication method*	**SFTP Support** : cochez cette case pour établir une méthode d'authentification. **Authentication method** : sélectionnez la méthode d'authentification à partir de la liste déroulante.

	Files	Cliquez sur le bouton **[+]** pour ajouter une ligne puis remplissez les colonnes. **Filemask** : noms et chemins d'accès aux fichiers à transférer. **New name** : nom à donner au fichier FTP après le transfert.
	Connection Mode	Sélectionnez votre mode de connexion. **Passive** : vous déterminez le port de connexion à utiliser pour permettre le transfert des données. **Active** : le serveur FTP détermine le port de connexion à utiliser pour permettre le transfert des données.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
Advanced settings	*Use Socks Proxy*	Cochez cette case si vous souhaitez utiliser un proxy, puis paramétrez les champs **Host**, **Port**, **User** et **Password** du proxy.
	Ignore Failure At Quit (FTP)	Cochez cette case afin d'ignorer les erreurs dues à la fermeture du FTP ou de la bibliothèque.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme sous-Job a un seul composant mais il peut aussi être utilisé comme composant de sortie ou de fin.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Copier des fichiers sur un serveur FTP distant, page 1053* .

tJMSInput

Propriétés du tJMSInput

Famille de composant	Internet	
Fonction	Ce composant crée une interface entre une application Java et un système Middleware orienté Message.	
Objectif	Le **tJMSInput** permet de mettre en place des interactions entre composants applicatifs dans un cadre faiblement couplé, asynchrone et fiable via un serveur JMS.	
Basic settings	*Module List*	Sélectionnez dans cette liste la bibliothèque à utiliser.
	Context Provider	Saisissez l'URL de contexte, par exemple "com.tibco.tibjms.naming.TibjmsInitialContextFactory". Attention cependant, la syntaxe peut varier selon le serveur JMS utilisé
	Server URL	Saisissez l'URL du serveur en respectant la syntaxe, par exemple "tibjmsnaming://localhost:7222".
	Connection Factory JDNI Name	Saisissez le nom JDNI.
	Use Specified User Identity	Si vous devez vous identifier, cochez la case et saisissez votre nom d'utilisateur et votre mot de passe.
	Message Type	Sélectionnez le type de message, Topic ou Queue.
	Message From	Saisissez la source du message, telle qu'attendue par le serveur. Celle-ci peut être composé du type et du nom de la source, par exemple : queue/A ou topic/testtopic. Notez que ce champ est sensible à la casse.
	Timeout for Next Message (in sec)	Renseignez le délai (en secondes) avant de passer au message suivant.
	Maximum Messages	Saisissez le nombre maximal de messages à traiter.
	Message Selector Expression	Spécifiez votre filtre.
	Processing Mode	Sélectionnez le mode de traitement des messages: Raw Message : Message brut. Message Content :
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma du composant **tJMSInput** est en lecture seule. Il est composé d'une colonne : **Message**
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé en tant que composant de début. Il requiert un composant de sortie.	

Limitation	Assurez-vous que le serveur JMS correspondant est lancé.

Scénario associé :

Pour un scénario associé, consultez le *Scénario : Communication asynchrone via un serveur MOM, page 1069*.

tJMSOutput

Propriétés du tJMSOutput

Famille de composant	Internet	
Fonction	Ce composant crée une interface entre une application Java et un système Middleware orienté Message.	
Objectif	Le **tJMSOutput** permet de mettre en place des interactions entre composants applicatifs dans un cadre faiblement couplé, asynchrone et fiable via un serveur JMS.	
Basic settings	*Module List*	Sélectionnez dans cette liste la bibliothèque à utiliser.
	Context Provider	Saisissez l'URL de contexte, par exemple "com.tibco.tibjms.naming.TibjmsInitialContextFactory". Attention cependant, la syntaxe peut varier selon le serveur JMS utilisé
	Server URL	Saisissez l'URL du serveur en respectant la syntaxe, par exemple "tibjmsnaming://localhost:7222".
	Connection Factory JDNI Name	Saisissez le nom JDNI.
	Use Specified User Identity	Si vous devez vous identifier, cochez la case et saisissez votre nom d'utilisateur et votre mot de passe.
	Message Type	Sélectionnez le type de message, Topic ou Queue.
	To	Saisissez la cible du message, telle qu'attendue par le serveur.
	Processing Mode	Sélectionnez le mode de traitement des messages: **Raw Message** : Message brut. **Message Content** : contenu du message.
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma du composant **tJMSInput** est en lecture seule. Il est composé d'une colonne : **Message**
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé en tant que composant de sortie. Il requiert un composant d'entrée.	
Limitation	Assurez-vous que le serveur JMS correspondant est lancé.	

Scénario associé :

Pour un scénario associé, consultez le *Scénario : Communication asynchrone via un serveur MOM, page 1069*.

tMicrosoftMQInput

Propriétés du tMicrosoftMQInput

Famille de composant	Internet/MOM et JMS	
Fonction	Le composant **tMicrosoftMQInput** récupère le premier message d'une file d'attente de message Microsoft donnée (le composant supporte uniquement le type String).	
Objectif	Ce composant vous permet de récupérer les messages un par un dans l'ordre de leur ID, dans la file d'attente de message Microsoft. Chaque exécution récupère un seul message.	
Basic settings	*PROPERTY*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Queue	Saisissez le nom de la file d'attente à partir de laquelle vous souhaitez récupérer les messages.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du Job ainsi qu'au niveau de chaque composant..
Utilisation	Ce composant est généralement utilisé en tant que composant de début dans un Job ou un sous-job. Il doit être lié à un composant de sortie.	
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : Run if, On Subjob Ok, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Iterate **Trigger** : Run if, On Subjob Ok, On Component Ok, On Component Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** Talend Open Studio.
Limitation	Ce composant ne supporte que le type String.	

Scénario : Ecrire et récupérer des messages dans une file d'attente de message Microsoft

Ce scénario se compose de deux Jobs. Le premier Job poste des messages dans une file de message Microsoft et le second Job récupère les messages du serveur.

Dans le premier Job, un message de type string est créé via un **tRowGenerator** et posté dans une file d'attente de message Microsoft à l'aide d'un composant **tMicrosoftMQOutput**. Un composant **tLogRow** intermédiaire affiche le flux passé d'un composant à l'autre.

- Déposez de la **Palette** dans l'espace de modélisation graphique un **tRowGenerator**, un **tLogRow** et un **tMicrosoftMQOutput**.
- Cliquez-droit sur le composant **tRowGenerator** afin d'ouvrir son menu contextuel.
- Dans ce menu, sélectionnez **Row** > **Main** afin de relier ce composant au **tLogRow**.
- Effectuez la même opération pour relier le **tLogRow** au **tMicrosoftMQOutput**.
- Double-cliquez sur le **tRowGenerator** afin d'ouvrir son éditeur.

- Dans cet éditeur, cliquez sur le bouton **[+]** pour ajouter trois lignes au schéma.
- Dans la colonne **Column**, saisissez un nouveau nom pour chaque ligne, afin de les renommer. Ici, saisissez respectivement *ID*, *Name* et *Address*.
- Dans la colonne **Type**, sélectionnez **Integer** pour la ligne *ID* dans la liste déroulante et laissez les autres lignes en **String**.
- Dans la colonne **Functions**, sélectionnez `random` pour la ligne *ID*, `getFirstName` pour la ligne *Name* et `getUsCity` pour la ligne *Address*.
- Dans le champ **Number of Rows for RowGenerator**, saisissez *12* afin de limiter le nombre de lignes à générer.
- Cliquez sur **OK** pour valider vos modifications.

Lors d'une utilisation réelle, vous devrez utiliser un composant d'entrée pour charger les données qui vous intéressent, au lieu du **tRowGenerator**.

- Double-cliquez sur le composant **tMicrosoftMQOutput** afin d'ouvrir sa vue **Component**.

- Dans le champ **Host**, saisissez l'adresse de l'hôte. Dans cet exemple, saisissez *localhost*.
- Dans le champ **Queue**, saisissez le nom de la file de messages dans laquelle vous souhaitez écrire le message. Dans cet exemple, nommez-la *AddressQueue*.
- Dans le champ **Message column (String Type)**, sélectionnez *Address* dans la liste déroulante afin de déterminer le corps du message à écrire.
- Appuyez sur **F6** pour exécuter le Job.

```
Starting job tMicrosoftMQ at 14:03 18/11/2010.

[statistics] connecting to socket on port 3852
[statistics] connected
Queue open failure: Cannot open queue. (hr=MQ_ERROR_QUEUE_NOT_FOUND)
OpenQueueWithAccess (DIRECT=OS:localhost\private$\AddressQueue)
open: fmtname(DIRECT=OS:localhost\private$\AddressQueue) accessmode(1)
sharemode(0)
..............................................................................
.............................................attempting to create queue with
name= '.\private$\AddressQueue', label='Created by
org.talend.msmq.MsmqUtil.java'
OpenQueueWithAccess (DIRECT=OS:.\private$\AddressQueue)
open: fmtname(DIRECT=OS:.\private$\AddressQueue) accessmode(1) sharemode(0)
open: fmtname(DIRECT=OS:.\private$\AddressQueue) accessmode(2) sharemode(0)
7|Theodore|Atlanta
8|Andrew|Saint Paul
7|Benjamin|Oklahoma City
3|Abraham|Providence
4|Rutherford|Lincoln
6|Chester|Little Rock
11|Richard|Lincoln
8|Bill|Boston
9|Herbert|Jackson
10|James|Honolulu
5|Jimmy|Montpelier
9|Woodrow|Nashville
[statistics] disconnected
Job tMicrosoftMQ ended at 14:03 18/11/2010. [exit code=0]
```

Vous pouvez voir que la file a été créée automatiquement et que les messages ont été écrits.

Créez ensuite le second Job pour récupérer le premier message de la file d'attente.

- Déposez un **tMicrosoftMQInput** et un **tLogRow** de la **Palette** dans l'espace de modélisation graphique.
- Reliez ces deux composants à l'aide d'un lien **Row** > **Main**.
- Double-cliquez sur le **tMicrosoftMQInput** afin d'ouvrir sa vue **Component**.

- Dans le champ **Host**, saisissez le nom ou l'adresse de l'hôte. Ici, saisissez *localhost*.
- Dans le champ **Queue**, saisissez le nom de la file d'attente de message à aprtir de laquelle vous souhaitez récupérer le message. Dans cet exemple, saisissez *AddressQueue*.
- Appuyez sur **F6** pour exécuter le Job.

```
Starting job MQInput at 16:57 18/11/2010.

[statistics] connecting to socket on port 3719
[statistics] connected
OpenQueueWithAccess
(DIRECT=OS:localhost\private$\AddressQueue)
open:
fmtname(DIRECT=OS:localhost\private$\AddressQueue)
accessmode(1) sharemode(0)
open:
fmtname(DIRECT=OS:localhost\private$\AddressQueue)
accessmode(2) sharemode(0)
Atlanta
[statistics] disconnected
Job MQInput ended at 16:57 18/11/2010. [exit code=0]
```

Le corps du message *Atlanta* récupéré de la file d'attente s'affiche dans la console.

tMicrosoftMQOutput

Propriétés du tMicrosoftMQOutput

Famille de composant	Internet/MOM and JMS	
Fonction	Le composant **tMicrosoftMQOutput** écrit une colonne définie d'un flux d'entrée de données dans une file de messages Microsoft (le composant supporte uniquement le type String).	
Objectif	Ce composant permet d'écrire des messages dans une file d'attente de messages Microsoft.	
Basic settings	*PROPERTY*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Host	Saisissez le nom de l'hôte ou l'adresse IP du serveur hôte.
	Queue	Saisissez le nom de la file de messages dans laquelle vous souhaitez écrire un message. Cette file peut être créée automatiquement à la volée si elle n'existe pas.
	Message column	Sélectionnez le message à écrire dans la file de messages Microsoft. Le message sélectionné doit être de type String.
Utilisation	Ce composant doit être lié à un composant d'entrée ou intermédiaire.	
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : Run if, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject, Iterate **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	Le message à écrire ne peut être null.	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Ecrire et récupérer des messages dans une file d'attente de message Microsoft, page 1064*

tMomInput

Propriétés du tMomInput

Famille de composant	Internet	
Fonction	Ce composant récupère un message d'une file d'attente d'un système Middleware orienté Message et le passe au composant suivant.	
Objectif	Le **tMomInput** permet de mettre en place des communications asynchrones via un serveur MOM.	
Basic settings	*MQ Server*	Sélectionnez dans cette liste le serveur MOM à utiliser. Les paramètres requis diffèrent légèrement en fonction du serveur sélectionné.
	Host/Port	Renseignez le nom de l'hôte (Host) et l'adresse IP du serveur MOM ainsi que le Port.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma du composant **tMomInput** est en lecture seule. Il est composé de deux colonnes : **From** et **Message**
JBoss Messaging	*Keep listening*	Cochez cette case pour consulter en continu le serveur MOM pour récupérer les nouveaux messages.
	Message From	Saisissez la source du message, telle qu'attendue par le serveur. Celle-ci peut être composé du type et du nom de la source, par exemple : queue/A ou topic/testtopic. Notez que ce champ est sensible à la casse.
	Message Body Type	Sélectionnez le type de message : **Text**, **Bytes** ou **Map**.
Websphere	*Channel*	La valeur par défaut est Channel.
	Queue Manager	Renseignez le gestionnaire de file(s) d'attente.
	Message Queue	Saisissez le nom de la file d'attente.
	Is using message id to fetch	Cochez cette case pour appliquer un filtre portant sur les messages id, filtre que vous déterminerez dans le champ **Message id**.
	Commit (delete message after reading from the queue)	Cochez cette case afin de forcer le commit après lecture de chaque message dans la file d'attente.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé en tant que composant de début. Il requiert un composant de sortie.	
Limitation	Assurez-vous que le serveur JBoss ou Websphere correspondant est lancé.	

Scénario : Communication asynchrone via un serveur MOM

Ce scénario est composé de deux Jobs. Le premier permet de poster des messages sur la file d'attente du serveur JBoss et le deuxième recupère les messages de ce serveur.

Dans le premier Job, un message composé de chaînes de caractères est créé à l'aide du composant **tRowGenerator** puis transmis au serveur JBoss à l'aide du composant **tMomOutput**. Un composant intermédiaire, le composant **tLogRow**, permet d'afficher le flux transmis.

- Cliquez et déposez ces trois composants dans votre premier Job et connectez-les via le clic-droit avec des liens de type **Main** row.
- Double-cliquez sur le composant **tRowGenerator** pour configurer le schéma des données qui seront générées de manière aléatoire.

- Créez une seule colonne appelée *message*. Elle correspond au message qui sera mis sur la file d'attente du MOM.
- Cette colonne est de type **String** et peut contenir une valeur nulle. Pour produire les données, utilisez une fonction prédéfinie permettant de concaténer de manière aléatoire les caractères ascii choisis pour former la chaîne de 6 caractères. Cette fonction est la suivante : `getAsciiRandomString`. (version Java). Cliquez sur le bouton d'aperçu pour afficher un échantillon au hasard des données générées.
- Paramétrez le champ **Number of rows to be generated** permettant de définir le nombre de lignes générée à *10*.
- Cliquez sur **OK** pour valider.
- Le composant **tLogRow** est uniquement utilisé pour afficher un état intermédiaire des données manipulées. Dans cet exemple, aucune configuration spécifique n'est nécessaire.
- Puis sélectionnez le composant **tMomOutput**.

- Dans cet exemple, le serveur **MQ server** à utiliser est de type **JBoss**.
- Dans les champs **Host** et **Port**, renseignez les informations de connexion correspondantes.
- Sélectionnez le type de message dans la liste déroulante **Message type**, il peut être de type **Queue** ou **Topic**. Pour cet exemple, sélectionnez le type **Queue** dans la liste.
- Dans le champ **To**, saisissez les informations source du message strictement telles qu'elles sont attendues par le serveur. Il doit correspondre au type de message que vous avez sélectionné dans la liste déroulante Message Type, notamment : `queue/A`.

  Le nom du message est sensible à la casse, ainsi `queue/A` et `Queue/A` sont différents.

- Puis cliquez sur **Sync Columns** pour récupérer le schéma du composant précédent. Le schéma étant en lecture seule, il ne peut être modifié. Les données envoyées sur le serveur MQ proviennent de la première colonne rencontrée du schéma.
- Exécutez le Job et, grâce au composant tLogRow, observez dans la console le flux de données transmis.

```
Starting job Mominput at 17:46 14/09/2007.
c8X5GC
lEhC41
opGYqP
x6gOt8
ESknZp
tXU2ET
6H7Nwl
8UA9eM
2sebwV
rfPZAP
Job Mominput ended at 17:46 14/09/2007. [exit code=0]
```

Puis paramétrez le deuxième Job pour récupérer les messages de la file d'attente du serveur MOM.

- Cliquez et déposez un composant **tMomInput** (dans la famille **Internet** de la **Palette**) et un composant **tLogRow** pour afficher les messages récupérés.
- Sélectionnez le **tMomInput** et définissez ses paramètres.

- Sélectionnez le serveur **MQ server** dans la liste. Dans cet exemple, un serveur de message JBoss est utilisé.
- Renseignez les informations **Host** et **Port** du serveur.
- Renseignez les paramètres des champs **Message From** et **Message Type** afin qu'ils correspondent à la source et au type de message attendus par le serveur de message.
- Le schéma dans le champ **Schema** est en lecture seule et est composé de deux colonnes : **From** et **Message**.
- Cochez la case **Keep listening** et paramétrez la fréquence de vérification à *5* secondes.

  Lorsque vous utilisez l'option **Keep Listening**, vous devez cliquez sur le bouton **Kill** pour stopper le Job.

- Laissez les paramètres par défaut du composant **tLogRow**.
- Enregistrez le Job et exécutez-le (si vous le lancez pour la première fois ou si vous l'avez stopper au cours d'un lancement précédent).

```
Starting job momoutput at 17:47 14/09/2007.
[statistics] connecting to socket on port 3414
[statistics] connected
Ready to receive message
Waiting...
queue/A|c8X5GC
queue/A|1EhC41
queue/A|opGYqP
queue/A|x6gOt8
queue/A|ESknZp
queue/A|tXU2ET
queue/A|6H7Nwl
queue/A|8UA9eM
queue/A|2sebwV
queue/A|rfPZAP
Job momoutput ended at 17:47 14/09/2007. [exit code=(
```

Les messages récupérés sur le serveur sont affichés dans la console.

tMomMessageIdList

Propriétés du tMomMessageIdList

Famille du composant	Internet	
Fonction	Le composant **tMomMessageIdList** récupère la liste des ID des messages à partir de la file d'attente d'un système Middleware orienté Message et la passe au composant suivant.	
Objectif	Le **tMomMessageIdList** permet d'itérer les ID de certains messages. Il est généralement utilisé avec le **tMomInput**. Pour plus d'informations, consultez *tMomInput, page 1068*.	
Basic settings	*MQ Server*	Sélectionnez dans cette liste le serveur MOM à utiliser. Les paramètres requis diffèrent légèrement en fonction du serveur sélectionné.
	Host/Port	Renseignez le nom de l'hôte (Host) et l'adresse IP du serveur MOM ainsi que le Port.
Websphere	*Channel*	La valeur par défaut est Channel
	Queue Manager	Renseignez le gestionnaire de file(s) d'attente.
	Message Queue	Saisissez le nom de la file d'attente.
Utilisation	Ce composant est généralement utilisé en tant que composant de début. Il requiert un composant de sortie.	
Limitation	Assurez-vous que le serveur Websphere correspondant est lancé.	

Scénario associé

Pour un scénario associé, consultez le composant *tMomInput, page 1068*.

tMomOutput

Propriétés du tMomOutput

Famille de composant	Internet	
Fonction	Dépose un message sur la file d'attente d'un système Middleware orienté message afin qu'il soit récupéré de manière asynchrone.	
Objectif	Le **tMomOutput** permet de mettre en place des communications asynchrones via un serveur MOM.	
Basic settings	*MQ Server*	Sélectionnez dans la liste le serveur MOM à utiliser. En fonction du serveur sélectionné, les paramètres diffèrent légèrement.
	Host/Port	Renseignez le nom de l'hôte ou l'adresse IP du serveur MOM dans le champ Host, ainsi que le Port.
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Lors de l'utilisation du composant **tMomOutput**, le schéma est en lecture seule mais changera en fonction du schéma d'entrée. Le serveur attend uniquement un schéma à une colonne pour contenir les **Messages**.
JBoss Messaging	*To*	Saisissez la destination du message, telle qu'elle est attendue par le serveur ; le type est le nom du dossier cible doivent être indiqués, par exemple : queue/A ou topic/testtopic Notez que ce champ est sensible à la casse.
	Message Type	Sélectionnez le type de message : **topic** ou **queue**.
Websphere	*Channel*	La valeur par défaut est Channel.
	Queue Manager	Renseignez le gestionnaire de file(s) d'attente.
	Message Queue	Saisissez le nom de la file d'attente.
Utilisation	Ce composant nécessite un composant d'entrée ou un composant intermédiaire.	
Limitation	Assurez-vous que le serveur JBoss ou Websphere correspondant soit bien lancé.	

Scénario associé

Pour un exemple d'utilisation du tMomOutput, consultez *tMomInput, page 1068*.

tPOP

Propriétés du tPOP

Famille du composant	Internet	
Fonction	Le composant **tPOP** récupère un ou plusieurs e-mails à partir d'un protocole POP3 ou IMAP.	
Objectif	Le composant **tPOP** utilise le protocole POP ou IMAP pour se connecter au serveur de messagerie spécifié. Il récupère ensuite un ou plusieurs e-mails et écrit les informations récupérées dans un ou plusieurs fichiers définis. Les paramètres avancés (**Advanced settings**) permettent d'appliquer un ou plusieurs filtres à la sélection.	
Basic settings	*Host*	Renseignez le nom de l'hôte ou l'adresse IP du serveur de messagerie auquel vous voulez vous connecter.
Java uniquement	*Port*	Renseignez le numéro du port du serveur de messagerie.
	Username et *Password*	Renseignez les informations d'authentification à votre compte de messagerie : **Username** : entrez l'identifiant que vous utilisez pour accéder à votre boîte mail. **Password** : entrez le mot de passe de connexion à votre boîte mail.
	Output directory	Renseignez le chemin d'accès au dossier dans lequel vous voulez stocker les e-mails récupérés à partir de votre messagerie, ou cliquez sur **[...]** pour parcourir vos dossiers jusqu'à l'emplacement souhaité.
	Filename pattern	Appuyez sur **Ctrl+Espace** pour afficher la liste des modèles prédéfinis ou définissez manuellement la syntaxe du nom des fichiers cible. Les données des e-mails récupérés à partir du serveur de messagerie sont écrites dans ces fichiers.
	Retrieve all emails?	Par défaut, tous les messages présents sur le serveur de messagerie spécifié sont récupérés. Pour ne récupérer qu'un nombre défini de messages, décochez cette case et renseignez dans le champ **Number of emails to retrieve** : le nombre d'e-mails à récupérer. La collecte des e-mails se fait du plus récent au plus ancien.
	Delete emails from server	Cochez cette case si vous ne souhaitez pas conserver sur le serveur les e-mails récupérés. Pour les serveurs Gmail, cette option ne fonctionne pas avec le protocole POP3. Sélectionnez le protocole IMAP et vérifiez que le compte Gmail est configuré pour utiliser ce protocole.

Java uniquement	*Choose the protocol*	Sélectionnez à partir de la liste déroulante le protocole à utiliser pour récupérer les e-mails. Le protocole dépend de celui utilisé par votre messagerie. Si vous choisissez le protocole imap, vous aurez la possibilité de spécifier le dossier contenant les e-mails que vous souhaitez récupérer.
Java uniquement	*Use SSL*	Cochez cette case si votre serveur de messagerie utilise ce protocole de sécurisation des échanges. Cette option est obligatoire pour les utilisateurs de messagerie Gmail.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Java uniquement	*Filter*	Cliquez sur le bouton **[+]** pour ajouter des lignes de filtre afin de ne récupérer qu'une sélection d'e-mails :
		Filter item : sélectionnez l'un des types de filtre suivants à partir de la liste déroulante : **From** : les messages sont filtrés en fonction de l'adresse e-mail de provenance. **To** : les messages sont filtrés en fonction de l'adresse e-mail de destination. **Subject** : les messages sont filtrés en fonction du sujet. **Before date** : les messages sont filtrés en fonction de la date d'émission ou de réception ; ils doivent être antérieurs à la date indiquée. **After date** : les messages sont filtrés en fonction de la date d'émission ou de réception ; ils doivent être postérieurs à la date indiquée.
		Pattern : appuyez sur **Ctrl+Espace** pour afficher la liste des valeurs disponibles et sélectionnez la valeur à utiliser pour chacun des filtres.
Java uniquement	*Filter condition relation*	Sélectionnez le type de relation à appliquer entre les filtres : **and** : les conditions déterminées par les filtres s'ajoutent les unes aux autres, la recherche est plus restrictive. **or** : les conditions déterminées par les filtres sont indépendantes les unes des autres, la recherche est plus large.
Utilisation	Ce composant ne traite pas de flux de données, il peut donc être utilisé seul.	
Limitation	n/a	

Scénario : Récupérer une sélection d'e-mails à partir d'un compte de messagerie

Ce scénario Java à un seul composant a pour but de récupérer un nombre prédéfini d'e-mails à partir d'un compte de messagerie.

- A partir de la **Palette**, cliquez-déposez un composant **tPOP** dans l'éditeur graphique,

- Double-cliquez sur le **tPOP** pour afficher la vue **Basic settings** et paramétrer le composant.
- Renseignez les informations de connexion au serveur de messagerie : le nom de l'hôte ou son adresse IP entre guillemets dans le champ **Host** et le numéro de port dans le champ **Port**.
- Saisissez ensuite les informations d'authentification de votre compte de messagerie. Dans le champ **Username**, saisissez entre guillemets votre identifiant et dans le champ **Password**, saisissez votre mot de passe. Dans cet exemple, le serveur de messagerie s'appelle *Free*.

- Dans le champ **Output directory**, sélectionnez manuellement le chemin d'accès ou cliquez sur **[...]** pour sélectionner le dossier de destination des e-mails récupérés sur le serveur.
- Dans le champ **Filename pattern**, définissez la syntaxe de nommage des fichiers de destination ou appuyez sur **Ctrl+Espace** pour afficher la liste des modèles prédéfinis. Dans cet exemple, la syntaxe est la suivante :
  `TalendDate.getDate("yyyyMMdd-hhmmss") + "_" + (counter_tPOP_1 + 1) + ".txt"`
  Les fichiers de destination, enregistrés au format texte, sont définis par leur date, heure et ordre de collecte.
- Décochez la case **Retrieve all emails?** pour ne récupérer qu'un nombre défini d'e-mails. Dans cet exemple, 10 messages sont récupérés.
- Cochez la case **Delete emails from server** pour supprimer les e-mails du serveur une fois qu'ils ont été récupérés en local.
- Sélectionnez un type de protocole à partir de la liste déroulante **Choose the protocol**. Le choix du protocole dépend de celui utilisé par votre fournisseur de messagerie. Certains fournisseurs comme *Gmail* utilisent les deux protocoles. Dans cet exemple, le protocole sélectionné est *pop3*.
- Enregistrez le Job puis appuyez sur **F6** pour l'exécuter.

Le composant **tPOP** a ainsi récupéré les 10 derniers messages du compte de messagerie spécifié.

Pour chaque e-mail récupéré, un fichier .txt a été créé dans le dossier *tPOP*. Chaque fichier contient les métadonnées d'envoi du message (expéditeur, destinataire, objet du message) ainsi que les données propres au message.

tRSSInput

Propriétés du tRSSInput

Famille du composant	Internet	
Fonction	Le composant **tRSSInput** lit des flux RSS en utilisant des URLs.	
Objectif	Le **tRSSInput** permet de suivre les billets postés sur certains sites en particulier afin d'en rassembler et organiser les informations rapidement et facilement pour votre utilisation.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Dans le contexte d'utilisation du **tRSSInput**, le schéma est composé de quatre colonnes : **TITLE**, **DESCRIPTION**, **PUBDATE** et **Link**. Les paramètres de titre sont en lecture seule, contrairement à leur type et leur longueur.
	RSS URL	Saisissez l'URL du flux RSS à lire.
	Read articles from	Lorsque cette case est cochée, le composant **tRSSInput** lit les articles du flux RSS à partir de la date renseignée en cliquant sur le bouton **[...]** situé à côté du champ **date time**.
	Max number of articles	Lorsque cette case est cochée, le composant **tRSSInput** lit autant d'articles que le nombre spécifié dans le champ **max amount**.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et les lignes contenant les erreurs seront ignorées.
Utilisation	Ce composant est généralement utilisé en tant que composant de début. Il requiert un composant de sortie.	
Limitation	n/a.	

Scénario : Récupérer les billets fréquemment mis à jour sur un blog

Ce scénario Java à deux composants a pour but de récupérer les billets qui sont souvent mis à jour sur un blog, à partir d'un flux RSS local **Talend**, en utilisant le composant **tRSSInput**.

- A partir de la **Palette**, cliquez-déposez les composants suivants dans l'éditeur graphique : le **tRSSInput** et le **tLogRow**.
- Cliquez-droit sur le premier pour le connecter au second à l'aide d'un lien de type **Row Main**.

- Dans l'éditeur graphique, sélectionnez le **tRSSInput**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tRSSInput**.

- Configurez le champ **Schema Type** en mode **Built-In** puis cliquez sur le bouton **[...]** situé à côté du champ **Edit Schema** pour changer le type et la longueur des paramètres du schéma si nécessaire.
- Cliquez sur **OK** pour fermer la boîte de dialogue.

Le schéma du **tRSSInput** est composé de quatre colonnes, *TITLE*, *Description*, *PUBDATE*, et *LINK*. Il est en lecture seule, excepté pour les paramètres de type et de longueur.

- Dans l'onglet **Basic settings** du composant **tRSSInput**, saisissez l'URL d'accès au flux RSS. Dans ce scénario, le **tRSSInput** est associé au flux RSS **Talend** : *http://feeds.feedburner.com/Talend*.

- Cochez/décochez les autres cases selon vos besoins. Dans ce scénario, on veut afficher les informations de deux articles datés du 20 juillet 2008.
- Dans l'espace graphique, sélectionnez le **tLogRow** puis cliquez sur la vue **Component** pour en définir la configuration de base. Pour plus d'informations sur les propriétés du composant **tLogRow**, consultez les *Propriétés du tLogRow, page 1163*.
- Enregistrez le Job puis appuyez sur **F6** pour l'exécuter.

```
Starting job RSSInput at 15:17 07/08/2008.
.-------------------------------------------------------------------------
|                                    #1. tLogRow_1
+-------------+-----------------------------------------------------------
| key         | value
+-------------+-----------------------------------------------------------
| TITLE       | Welcoming Jean-Luc Solans
| DESCRIPTION | Jean-Luc Solans joins Talend as VP of Strategy and Busines
| PUBDATE     | 24 Jul 2008 19:40:11 GMT
| LINK        | http://feeds.feedburner.com/~r/Talend/~3/344920575/
+-------------+-----------------------------------------------------------

.-------------------------------------------------------------------------
|                                  #2. tLogRow_1
+-------------+-----------------------------------------------------------
| key         | value
+-------------+-----------------------------------------------------------
| TITLE       | Talend Open Profiler gets rave reviews
| DESCRIPTION | An interview and a product review of Talend Open Profiler.
| PUBDATE     | 23 Jul 2008 21:12:53 GMT
| LINK        | http://feeds.feedburner.com/~r/Talend/~3/343928056/
+-------------+-----------------------------------------------------------

Job RSSInput ended at 15:17 07/08/2008. [exit code=0]
```

Le composant **tRSSInput** a accédé au flux RSS du site de **Talend** à votre place et en a organisé les informations pour vous.

Deux billets sont affichés sur la console. Chacun d'eux a ses propres titre, description, date de publication et URL du flux RSS correspondante. Les blogs présentent le dernier billet en premier et vous pouvez faire défiler vers le bas pour lire les billets les plus anciens.

tRSSOutput

Propriétés du tRSSOutput

Famille du composant	Internet	
Fonction	Le composant **tRSSOutput** écrit des fichiersXML RSS_Feed ou Atom_Feed.	
Objectif	Le **tRSSOutput** permet de créer des fichiers XML contenant des flux RSS ou Atom.	
Basic settings	*File Name*	Chemin d'accès ou nom du fichier XML en sortie. Consultez également la section *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
	Append	Cochez cette option pour ajouter des nouvelles lignes à la fin du fichier.
	Mode	Sélectionnez RSS ou ATOM selon le flux que vous souhaitez générer.
	Channel (en mode RSS)	Les informations à renseigner dans cette zone concernent l'intégralité des données saisies, du site, etc. plutôt que des éléments particuliers. **Title** : Saisissez un titre significatif. **Description** : Entrez une description du contenu qui vous semble suffisamment détaillée. **Publication date** : Renseignez la date adéquate. **Link** : Saisissez l'URL appropriée.
	Feed (en mode ATOM)	**Title** : Saisissez un titre significatif. **Link** : Saisissez l'URL appropriée. **Id** : Saisissez l'URL valide correspondant au champ **Link**. **Update date** : Saisissez la date de mise à jour. **Author name** : Saisissez le nom de l'auteur.
	Optionnal Channel Elements	Renseignez dans le tableau les informations relatives aux métadonnées du flux RSS. Cliquez sur le bouton **[+]** pour ajouter de nouvelles lignes : **Element Name** : nom de la métadonnée. **Element Value** : contenu de la métadonnée.

	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Dans le contexte d'utilisation du **tRSSInput**, le schéma est composé de quatre colonnes : **TITLE**, **DESCRIPTION**, **PUBDATE** et **Link**. Les paramètres de titre sont en lecture seule, contrairement à leur type et leur longueur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant doit obligatoirement être relié à un composant de début ou un composant intermédiaire.	
Limitation	n/a	

Scénario 1 : Créer un flux RSS et stocker les fichiers sur un serveur FTP

Ce scénario Java a pour but de :

- créer un flux RSS pour des fichiers que vous souhaitez partager avec d'autres personnes, et
- stocker l'ensemble des fichiers sur un serveur FTP.

Ce scénario permet d'écrire un fichier XML de flux RSS à partir d'une table MySQL contenant des informations concernant des livres. Il ajoute également des liens vers les fichiers stockés sur un serveur FTP au cas où les utilisateurs voudraient avoir accès aux fichiers dans leur intégralité.

- A partir de la **Palette**, cliquez-déposez les composants suivants dans l'éditeur graphique : le **tMysqlInput**, le **tRSSOutput** et le **tFTPPut**.
- Cliquez droit sur le **tMysqlInput** et connectez-le au **tRSSOutput** à l'aide d'un lien de type **Row Main**.
- Cliquez droit sur le **tMysqlInput** et connectez-le au **tFTPPut** à l'aide d'un lien de type **OnSubjobOk**.

- Dans l'espace graphique, sélectionnez le **tMysqlInput**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tMysqlInput**.

- Configurez le champ **Property type** en mode **Repository** puis cliquez sur le bouton **[...]** pour sélectionner la base de données adéquate dans la liste. Les détails de connexion ainsi que le schéma sont remplis automatiquement.
- Dans le champ **Table Name**, saisissez le nom de votre table ou cliquez sur le bouton **[...]** et sélectionnez le nom de la table à partir de la liste. Dans ce scénario, la table Mysql de données saisies s'appelle "rss_talend" et le schéma comporte quatre colonnes, *TITLE*, *Description*, *PUBDATE*, et *LINK*.
- Dans le champ **Query**, saisissez la requête de base de données en faisant particulièrement attention à ce que l'ordre des champs corresponde au schéma défini, ou cliquez sur **Guess Query**.
- Dans l'espace graphique, sélectionnez le composant **tRSSOutput**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tRSSOutput**.

- Dans le champ **File name**, utilisez le nom de fichier et son chemin d'accès définis par défaut ou parcourez vos dossiers pour définir votre propre fichier XML de sortie.
- Sélectionnez le type d'encodage à partir de la liste **Encoding Type**.
- Dans la zone **Mode**, sélectionnez **RSS**.
- Dans le panneau **Channel**, saisissez un titre, une description, une date de publication et un lien définissant l'ensemble de vos données saisies.
- Sélectionnez le type de schéma dans la liste **Schema Type** puis cliquez sur **Edit Schema** pour le modifier si nécessaire.

  Vous pouvez également cliquer sur **Sync Column** pour récupérer le schéma généré par le composant précédent.

- Enregistrez votre Job puis appuyez sur **F6** pour en exécuter la première partie.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
- <rss version="2.0">
  - <channel>
      <title>library index</title>
      <description>grouping new books</description>
      <pubdate>2008-08-16 14:55:29</pubdate>
      <link>http://feeds.feedburner.com/Talend</link>
    - <item>
        <title>Book 1</title>
        <description>Summary of book1</description>
        <pubdate>2008-01-02</pubdate>
        <link>ftp://localhost/file_1.txt</link>
      </item>
    - <item>
        <title>Book 2</title>
        <description>Summary of book2</description>
        <pubdate>2008-01-02</pubdate>
        <link>ftp://localhost/file_2.txt</link>
      </item>
    - <item>
        <title>Book 3</title>
        <description>Summary of book3</description>
        <pubdate>2008-01-02</pubdate>
        <link>ftp://localhost/file_3.txt</link>
      </item>
    </channel>
  </rss>
```


Le composant **tRSSOutput** a créé en sortie un flux RSS au format XML pour les fichiers définis.

Pour stocker l'intégralité des fichiers sur un serveur FTP :

- Dans l'espace graphique, sélectionnez le **FTPPut**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tFTPPut**.

- Saisissez le nom du serveur et le numéro de port dans les champs correspondants.
- Renseignez vos détails de connexion dans les champs **Username** et **Password**.
- Parcourez vos dossier jusqu'au répertoire local ou tapez manuellement le chemin d'accès dans le champ **Local directory**.
- Entrez les détails du serveur de répertoire distant.
- Sélectionnez le mode de transfert à partir de la liste **Transfer mode**.
- Dans le panneau **Files**, cliquez sur le bouton **[+]** pour ajouter des lignes puis remplissez les masques de fichiers de tous les fichiers à copier sur le répertoire distant. Dans ce scénario, les fichiers à enregistrer sur le serveur FTP sont tous des fichiers texte.
- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

Les fichiers définis sont copiés sur le serveur distant.

Scénario 2 : Créer un flux RSS contenant des métadonnées

Ce scénario Java à deux composants a pour but de créer un flux RSS contenant des métadonnées et de rediriger les informations obtenues dans un fichier XML de flux RSS en sortie.

- A partir de la **Palette**, cliquez-déposez les composants **tRSSInput** et **tRSSOutput** dans l'éditeur graphique.
- Cliquez droit sur le **tRSSInput** et connectez-le au **tRSSOutput** à l'aide d'un lien de type **Row Main**.

- Dans l'espace graphique, sélectionnez le **tRSSInput**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tRSSInput**.

- Définissez le type du schéma comme **Built-In**.

Si votre schéma est déjà stocké localement dans le **Repository**, sélectionnez l'option **Repository** dans le champ **Schema**, puis cliquez sur le bouton **[...]** à côté du champ **Edit Schema** pour afficher une boîte de dialogue dans lasquelle vous pouvez choisir les métadonnées appropriées. Pour plus d'informations concerenant les métadonnées, consultez *Gestion des métadonnées* dans le **Guide Utilisateur** **Talend Open Studio**.

- Cliquez sur le bouton **[...]** situé à côté du champ **Edit Schema** pour vérifier les paramètres du schéma si nécessaire.

Le schéma du **tRSSInput** est composé de quatre colonnes, *TITLE*, *DESCRIPTION*, *PUBDATE*, et *LINK* et il est en lecture seule.

- Cliquez sur **OK** pour fermer la boîte de dialogue.
- Dans l'espace graphique, double-cliquez sur le composant **tRSSOutput** pour ouvrir sa vue **Component** et définir sa configuration de base (**Basic settings**).

- Dans le champ **File name**, utilisez le nom de fichier de sortie et son chemin d'accès définis par défaut ou parcourez vos dossiers pour définir votre propre fichier XML de sortie.
- Sélectionnez le type d'encodage à partir de la liste **Encoding Type**.
- Dans la zone **Mode**, sélectionnez **RSS**.
- Dans le panneau **Channel**, saisissez un titre, une description, une date de publication et un lien définissant l'ensemble de vos données saisies.
- Dans le tableau **Optional Channel Element**, définissez les métadonnées de votre flux RSS. Dans ce scénario, le flux comprend deux métadonnées : *copyright*, à laquelle est associée la valeur *tos*, et *language*, à laquelle est associée la valeur *en_us*.
- Sélectionnez le type de schéma dans la liste **Schema Type** puis cliquez sur **Edit Schema** pour le modifier si nécessaire.

  Vous pouvez également cliquer sur **Sync Column** pour récupérer le schéma généré par le composant précédent.

- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

Les fichiers définis sont copiés dans le fichier XML de sortie et les métadonnées apparaissent sous le noeud *<channel>*, en dessous des informations de base du flux RSS.

Scénario 3 : Créer un fichier XML de flux ATOM

Ce scénario Java décrit un Job à deux composants, qui génère des données et les écrit dans un fichier XML de flux ATOM.

- Déposez les composants suivants de la **Palette** dans l'espace de modélisation graphique : un **tFixedFlowInput** du groupe **Misc** et un **tRSSOutput** du groupe **Internet**.
- Cliquez-droit sur le **tFixedFlowInput** et connectez-le au **tRSSOutput** à l'aide d'un lien **Row Main**.
- Une fenêtre pop-up s'affiche et vous demande si vous souhaitez passer le schéma du **tRSSOutput** au **tFixedFlowInput**, cliquez sur **Yes** (Oui).

- Dans l'espace de modélisation graphique, double-cliquez sur le **tFixedFlowInput** afin d'afficher sa vue **Component** et définir ses **Basic settings**.

- Laissez le type de **Schema** en **Built-in**. Cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour afficher le schéma importé du composant de sortie.
- Dans le champ **Number of rows**, laissez le paramètre par défaut, **1**, pour ne générer qu'une ligne de données.
- Dans la zone **Mode**, laissez sélectionnée l'option **Use Single Table** et remplissez le tableau **Values**. Notez que le champ **Column** du tableau **Values** est rempli par les colonnes du schéma défini dans le composant.
- Dans le champ **Value** du tableau **Values**, saisissez les données que vous souhaitez envoyer au composant suivant.
- Dans l'espace de modélisation, double-cliquez sur le composant **tRSSOutput** afin d'afficher sa vue **Component** et définir ses **Basic settings**.

- Cliquez sur le bouton **[...]** à côté du champ **File Name** afin de définir le répertoire et le nom du fichier XML de sortie.
- Dans la zone **Mode**, sélectionnez **ATOM** pour générer un fichier XML de flux ATOM.
- Dans la zone **Feed**, saisissez un titre, un lien, un ID, la date de mise à jour, et le nom de l'auteur, afin de définir vos données comme un tout.
- Sélectionnez le type de votre schéma, dans la liste **Schema Type** puis cliquez sur le bouton **[...]** du champ **Edit Schema** afin d'afficher le schéma et de le modifier si nécessaire.

Comme le format du flux ATOM est strict, des informations par défaut sont requises afin de créer le fichier XML. Le schéma du **tRSSOutput** comprend donc des colonnes par défaut contenant ces informations. Ces colonnes par défaut sont grisées pour indiquer qu'elles ne doivent pas être modifiées. Si vous choisissez de modifier le schéma du composant, le fichier XML ATOM créé ne sera pas valide.

- Save your Job and press **F6** to execute it.

Le composant **tRSSOutput** crée un flux de sortie ATOM au format XML.

tSCPClose

Propriétés du tSCPClose

Famille de composant	Internet/SCP	
Fonction	Le composant **tSCPClose** ferme la connexion au protocole entièrement crypté.	
Objectif	Ce composant permet de fermer une connexion au protocole SCP.	
Basic settings	*Component list*	S'il y a plus d'une connexion dans le Job en cours, sélectionnez le composant **tSCPConnection** dans la liste.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario associé

Ce composant est étroitement lié aux composants **tSCPConnection** et **tSCPRollback**. Il est généralement utilisé avec un composant **tSCPConnection** car il permet de fermer une connexion pour la transaction en cours.

Pour un scénario associé au composant **tSCPClose**, consultez *tMysqlConnection, page 471*..

tSCPConnection

Propriétés du tSCPConnection

Famille de composant	Internet/SCP	
Fonction	Le composant **tSCPConnection** ouvre une connexion SCP afin d'effectuer une transaction.	
Objectif	Le **tSCPConnection** permet d'ouvrir une connexion SCP afin d'effectuer des transferts sécurisés de fichiers en une seule transaction.	
Basic settings	*Host*	Adresse IP du serveur SCP.
	Port	Numéro du port d'écoute du serveur SCP.
	Username	Login de connexion au serveur SCP.
	Authentication method	Méthode d'authentification au serveur SCP.
	Password	Mot de passe de connexion au serveur SCP.
Utilisation	Ce composant est généralement utilisé comme sous-Job à un seul composant.	
Limitation	n/a	

Scénarios associés

Pour un scénario associé, consultez le *Scénario : Copier des fichiers sur un serveur FTP distant, page 1053*.

Pour un scénario associé utilisant un protocole différent, consultez le *Scénario : Récupérer des fichiers d'un serveur SCP distant, page 1098*

tSCPDelete

Propriétés du tSCPDelete

Famille de composant	Internet/SCP	
Fonction	Le composant **tSCPDelete** supprime les fichiers sélectionnés d'un serveur distant par le biais d'un protocole entièrement crypté.	
Objectif	Le **tSCPDelete** permet de supprimer un fichier d'un serveur SCP donné.	
Basic settings	*Host*	Adresse IP du serveur SCP.
	Port	Numéro du port d'écoute du serveur SCP.
	Username	Login de connexion au serveur SCP.
	Authentication method	Méthode d'authentification au serveur SCP.
	Password	Mot de passe de connexion au serveur SCP.
	Filelist	Nom ou chemin d'accès du fichier à supprimer.
Utilisation	Ce composant est généralement utilisé comme sous-Job à un seul composant mais il peut aussi être utilisé en association avec d'autres composants.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Récupérer des fichiers d'un serveur SCP distant, page 1098*.

Pour un scénario associé utilisant un protocole différent, consultez le *Scénario : Copier des fichiers sur un serveur FTP distant, page 1053*.

tSCPFileExists

Propriétés du tSCPFileExists

Famille de composant	Internet/SCP	
Fonction	Le composant **tSCPFileExists** vérifie si un fichier est présent sur le serveur distant, par le biais d'un protocole entièrement crypté.	
Objectif	Le **tSCPFileExists** permet de vérifier la présence d'un fichier sur le serveur SCP spécifié.	
Basic settings	*Host*	Adresse IP du serveur SCP.
	Port	Numéro du port d'écoute du serveur SCP.
	Username	Login de connexion au serveur SCP.
	Authentication method	Méthode d'authentification au serveur SCP.
	Password	Mot de passe de connexion au serveur SCP.
	Remote directory	Chemind'accès au répertoire du serveur distant.
	Filename	Nom du fichier à chercher.
Utilisation	Ce composant est généralement utilisé comme sous-Job à un seul composant mais il peut aussi être utilisé en association avec d'autres composants.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Récupérer des fichiers d'un serveur SCP distant, page 1098*.

Pour un scénario associé utilisant un protocole différent, consultez le *Scénario : Copier des fichiers sur un serveur FTP distant, page 1053*.

tSCPFileList

Propriétés du tSCPFileList

Famille de composant	Internet/SCP	
Fonction	Le composant **tSCPFileList** fait une opération d'itération sur les fichiers d'un répertoire du serveur distant, par le biais d'un protocole entièrement crypté.	
Objectif	Le **tSCPFileList** permet de lister les fichiers d'un serveur SCP spécifié.	
Basic settings	*Host*	Adresse IP du serveur SCP.
	Port	Numéro du port d'écoute du serveur SCP.
	Username	Login de connexion au serveur SCP.
	Authentication method	Méthode d'authentification au serveur SCP.
	Password	Mot de passe de connexion au serveur SCP.
	Command separator	Le caractère utilisé pour séparer les différentes commandes.
	Filelist	Nom ou chemin d'accès au répertoire contenant les fichiers à lister.
Utilisation	Ce composant est généralement utilisé comme sous-Job à un seul composant mais il peut aussi être utilisé en association avec d'autres composants.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Récupérer des fichiers d'un serveur SCP distant, page 1098*.

Pour un scénario associé utilisant un protocole différent, consultez le *Scénario : Copier des fichiers sur un serveur FTP distant, page 1053*.

tSCPGet

Propriétés du tSCPGet

Famille de composant	Internet/SCP	
Fonction	Le composant **tSCPGet** transfère les fichiers sélectionnés par le biais d'un protocole entièrement crypté via une connexion SCP.	
Objectif	Le **tSCPGet** permet de copier de fichiers du serveur SCP spécifié.	
Basic settings	*Host*	Adresse IP du serveur SCP.
	Port	Numéro du port d'écoute du serveur SCP.
	Username	Login de connexion au serveur SCP.
	Authentication method	Méthode d'authentification au serveur SCP.
	Password	Mot de passe de connexion au serveur SCP.
	Local directory	Chemin d'accès du répertoire de destination.
	Overwrite or Append	Liste des options disponibles pour le transfert des fichiers.
	Filelist	Nom ou chemin d'accès du ou des fichier(s) à copier.
Utilisation	Ce composant est généralement utilisé comme sous-Job à un seul composant mais il peut aussi être utilisé en association avec d'autres composants.	
Limitation	n/a	

Scénario : Récupérer des fichiers d'un serveur SCP distant

Ce scénario Java crée un Job à un seul composant qui récupère le fichier sélectionné d'un serveur SCP distant.

- Cliquez et déposez le composant **tSCPGet** dans l'éditeur graphique.
- Dans l'éditeur graphique, sélectionnez le **tSCPGet** et cliquez sur la vue **Component** pour paramétrer ses propriétés de base.

- Dans le champ **Host**, renseignez l'adresse IP du serveur SCP.
- Dans le champ **Port** ,renseignez le numéro du port d'écoute du serveur SCP.
- Dans les champs **Username** et **Password**, renseignez respectivement le login et le mot de passe de connexion au serveur SCP.
- Dans la liste **Authentication method**, sélectionnez la méthode d'authentification appropriée.
  Notez que les champs suivants changent en fonction de la méthode d'authentification sélectionnée. La méthode d'authentification *Password* est utilisée dans ce scénario.
- Dans le champ **Local directory**, renseignez le chemin d'accès au répertoire dans lequel sera copié le fichier.
- Dans la liste **Overwrite or Append**, sélectionnez l'action à effectuer.
- Dans la zone **Filelist**, cliquez sur le bouton **[+]** pour ajouter une ligne à la liste **Source** et renseignez le chemin d'accès au fichier sur le serveur SCP distant.
  Dans ce scénario, le fichier à copier du serveur SCP distant sur le disque local s'appelle *backport.pl*.
- Enregistrez le Job et appuyez sur **F6** l'exécuter.

Le fichier du serveur distant est copié sur le disque local.

tSCPPut

Propriétés du tSCPPut

Famille de composant	Internet/SCP	
Fonction	Le composant **tSCPPut** copie les fichiers sélectionnés sur un serveur SCP distant par le biais d'un protocole entièrement crypté.	
Objectif	Le **tSCPPut** permet de copier les fichiers dans le répertoire d'un serveur SCP distant.	
Basic settings	*Host*	Adresse IP d'un serveur SCP.
	Port	Numéro du port d'écoute du serveur SCP.
	Username	Login de connexion au serveur SCP.
	Authentication method	Méthode d'authentification au serveur SCP.
	Password	Mot de passe de connexion au serveur SCP.
	Remote directory	Chemin d'accès au répertoire de destination.
	Filelist	Nom ou chemin d'accès au(x) fichier(s) à copier.
Utilisation	Ce composant est généralement utilisé comme sous-Job à un seul composant mais il peut aussi être utilisé en association avec d'autres composants.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Récupérer des fichiers d'un serveur SCP distant, page 1098*.

Pour un scénario associé utilisant un protocole différent, consultez le *Scénario : Copier des fichiers sur un serveur FTP distant, page 1053*.

tSCPRename

Propriétés du tSCPRename

Famille de composant	Internet/SCP	
Fonction	Le composant **tSCPRename** renomme des fichiers sur un serveur SCP distant.	
Objectif	Le **tSCPRename** permet de renommer un ou des fichier(s) sur le serveur SCP sélectionné.	
Basic settings	*Host*	Adresse IP du serveur SCP.
	Port	Numéro du port d'écoute du serveur SCP.
	Username	Login de connexion au serveur SCP.
	Authentication method	Méthode d'authentification au serveur SCP.
	Password	Mot de passe de connexion au serveur SCP.
	File to rename	Nom ou chemin d'accès au fichier à renommer.
	Rename to	Saisissez le nouveau nom du fichier.
Utilisation	Ce composant est généralement utilisé comme sous-Job à un seul composant mais il peut aussi être utilisé en association avec d'autres composants.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Récupérer des fichiers d'un serveur SCP distant, page 1098*.

tSCPTruncate

Propriétés du tSCPRename

Famille de composant	Internet/SCP	
Fonction	Le composant **tSCPTruncate** supprime toutes les données d'un fichier via une connexion SCP.	
Objectif	Le **tSCPTruncate** permet de supprimer les données d'un ou plusieurs fichier(s) d'un serveur SCP.	
Basic settings	*Host*	Adresse IP du serveur SCP.
	Port	Numéro du port d'écoute du serveur SCP.
	Username	Login de connexion au serveur SCP.
	Authentication method	Méthode d'authentification au serveur SCP.
	Password	Mot de passe de connexion au serveur SCP.
	Remote directory	Chemin d'accès au fichier de destination.
	Filelist	Nom ou chemin d'accès au(x) fichier(s) à vider.
Utilisation	Ce composant est généralement utilisé en tant que sous-Job à un seul composant mais il peut aussi être utilisé en association avec d'autres composants.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Récupérer des fichiers d'un serveur SCP distant, page 1098*.

tSendMail

Propriétés du tSendMail

Famille de composant	Internet	
Fonction	Le composant **tSendMail** envoie des messages et toute autre pièce jointe aux destinataires spécifiés.	
Objectif	Le **tSendMail** permet de prévenir des destinataires d'un état particulier d'un Job ou de possibles erreurs.	
Basic settings	*To*	Adresse e-mail du destinataire principal.
	From	Adresse e-mail du serveur d'envoi.
	Show sender's name?	Cochez cette case pour définir le nom d'expéditeur à afficher dans les messages. **Sendername** : renseignez le nom de l'expéditeur.
	Cc	Adresse e-mail des destinataires en copie.
	Bcc	Adresse e-mail des destinataires en copie cachée.
	Subject	Objet du mail.
	Message	Contenu du message. Appuyez sur **Ctrl+Espace** pour afficher la liste des variables disponibles.
	Die if the attachment file doesn't exist	Décochez cette case pour que les messages soient envoyés même si aucune pièce jointe n'est attachée.
	Attachment	Cliquez sur le bouton **[+]** pour ajouter des lignes de filtre : **File** : masque du fichier ou chemin d'accès au fichier à joindre à l'e-mail.
	Other Headers	Cliquez sur le bouton **[+]** pour ajouter des lignes de filtre : **Key** : saisissez la clé de tout en-tête différent de l'en-tête standard. **Value** : saisissez la valeur correspondante à la clé.
	SMTP Host et Port	Adresse IP du serveur SMTP utilisé pour l'envoi d'e-mails.
	SSL Support	Cochez cette case pour authentifier le serveur auprès du client via un protocole sécurisé SSL.
	STARTTLS Support	Cochez cette case pour authentifier le serveur auprès du client via un protocole sécurisé STARTTLS.
	Importance	Sélectionnez dans la liste déroulante le niveau de priorité de votre message.

	Need authentication?	Cochez la case d'authentification et renseignez le nom de l'utilisateur (username) et son mot de passe (password), si cela est nécessaire pour accéder au service.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et les lignes contenant les erreurs seront ignorées.
Advanced settings	*MIME subtype from the 'text' MIME type*	Sélectionnez dans la liste déroulante la forme à attribuer au texte du message.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce component est généralement utilisé en tant que sous-Job mais peut aussi être utilisé comme composant de sortie ou de fin de Job. Il peut être connecté aux autres composants via des liens Row ou Iterate.	
Limitation	Notez que l'envoi d'e-mail requiert deux modules (Perl) différents suivant qu'il contient une pièce jointe ou non.	

Scénario : Envoyer un e-mail lors d'une erreur

Ce scénario est un Job composé de trois composants permettant d'envoyer un e-mail à des destinataires spécifiques lorsqu'une erreur se produit.

- A partir de la **Palette**, cliquez-déposez les composants suivants dans l'espace de modélisation : **tFileInputDelimited**, **tFileOutputXML** et **tSendMail**.
- Cliquez-droit sur le composant **tFileInputDelimited** et reliez-le au composant **tFileOutputXML** à l'aide d'un lien de type **Row** > **Main**.
- Cliquez-droit sur le composant **tFileInputDelimited** et reliez-le au composant **tSendMail** à l'aide d'un lien de type **Trigger** > **OnSubjobError**.
- Double-cliquez sur le composant **tFileInputDelimited** pour afficher l'onglet **Basic settings** du composant et configurer ses paramètres de base.

- A partir de la liste **Property type**, sélectionnez :
  -**Repository** si vous avez déjà stocké les métadonnées de votre fichier d'entrée dans le **Repository**, les champs suivants sont alors renseignés automatiquement avec les informations stockées, ou
  -**Built-in** si aucune métadonnée n'est stockée pour votre fichier d'entrée ; vous devez alors renseigner manuellement les champs suivants.
  Dans cet exemple, c'est le mode **Built-in** qui est utilisé.
- Dans le champ **File Name**, renseignez le chemin d'accès ou cliquez sur le bouton **[...]** pour parcourir vos dossiers jusqu'au fichier à traiter. Ce champ est obligatoire.
  Dans cet exemple, le fichier à traiter est *customer_mail.csv*.
- Définissez le séparateur de lignes dans le champ **Row separator** afin d'identifier la fin d'une ligne. Puis définissez le séparateur de champs dans **Field separator** pour délimiter les champs composant une ligne.
  Les séparateurs utilisés dans ce scénario sont des séparateurs de champs et de lignes classiques de fichier CSV.
- Dans ce scénario, l'en-tête (**Header**), le pied de page (**Footer**) et la limite de lignes lues (**Limit**) n'ont pas besoin d'être définis.
- Sélectionnez l'option **Built-in** à partir de la liste déroulante **Schema** et renseignez manuellement les informations concernant le fichier délimité dans les champs correspondants.
- Cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour afficher la boîte de dialogue permettant de définir la structure des données. Dans ce scénario, le schéma comporte quatre colonnes : *Id*, *Name*, *Address* et *IdState*.
- Cliquez sur le bouton **[+]** pour ajouter autant de lignes que nécessaire à la structure des données. Dans cet exemple, il n'y a qu'une seule colonne dans le schéma : *xmlStr*.
- Cliquez sur **OK** pour valider vos changements et fermer la boîte de dialogue.

Si vous avez préalablement stocké votre schéma dans le répertoire **Metadata** sous **File delimited**, sélectionnez l'option **Repository** dans la liste déroulante **Schema** puis cliquez sur le bouton **[...]** à côté du champ pour afficher la boîte de dialogue **[Repository Content]** à paritr de laquelle vous pouvez sélectionner un schéma pertinent. Cliquez sur **Ok** pour fermer la boîte de dialogue et remplir les champs automatiquement avec les métadonnées contenues dans le schéma.
Pour plus d'informations sur le stockage des schémas de métadonnées dans le **Repository**, consultez les sections *Configurer un schéma File Delimited* et *Déposer des composants à partir du répertoire Metadata* du **Guide Utilisateur** de **Talend Open Studio**.

Voir également : *Paramétrer un schéma built-in* et *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.

- Assurez-vous que la case **Skip empty rows** est cochée afin d'ignorer les lignes vierges.
- Double-cliquez sur le composant **tFileOutputXML** pour afficher l'onglet **Basic settings** du composant et configurer ses paramètres de base.

- Cliquez sur le bouton **[...]** à côté du champ **File Name** pour sélectionner le répertoire de destination du fichier et saisissez le nom du fichier. Dans ce scénario, appelez le fichier *customer_data.xml*.
- Dans le champ **Row tag**, saisissez entre guillemets le nom que vous souhaitez donner à la balise contenant les données.
- Cliquez sur **Edit schema** et assurez-vous que le schéma est récupéré à partir du schéma d'entrée. Si nécessaire, cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
- Double-cliquez sur le composant **tSendMail** pour afficher l'onglet **Basic settings** du composant et configurer ses paramètres de base.

- Saisissez les adresses e-mail des destinataires dans le champ **To** et les champs **Cc** et **Bcc** selon vos préférences, ainsi que celle de l'expéditeur dans le champ **From**.
- Cochez la case **Show sender's name?** puis renseignez le nom de l'expéditeur dans le champ **Sendername** pour que votre message ait une meilleure visibilité auprès de vos destinataires.
- Renseignez l'objet du message dans le champ **Subject**.
- Saisissez dans le champ **Message** le message contenant le code d'erreur produit à l'aide de la variable globale correspondante. Accédez à la liste des variables en appuyant sur **Ctrl+Espace**.
- Ajoutez des pièces jointes et des informations sur les en-têtes, si nécessaire, dans les champs respectifs **Attachments** et **Other headers**.
- Saisissez les informations SMTP dans les champs **SMTP host** et **SMTP port**.
- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

Dans ce scénario, le fichier contenant les données à transférer vers une sortie XML ne peut être trouvé. Le composant **tSendmail** s'exécute lorsque cette erreur est rencontrée et envoie un mail de notification au destinataire spécifié.

tSetKeystore

Propriétés du tSetKeystore

Famille de composant	Internet	
Fonction	Le composant **tSetKeystore** soumet des données d'authentification d'un TrustStore avec ou sans Keystore pour valider la connexion SSL.	
Objectif	Ce composant vous permet de définir le type des données d'authentification entre **PKCS 12** et **JKS**.	
Basic settings	*TrustStore type*	Sélectionnez le type of the TrustStore to be used. It may be **PKCS 12** or **JKS**.
	TrustStore file	Saisissez le chemin d'accès et le nom du fichier TrustStore, ou parcourez votre répertoire jusqu'au TrustStore contenant la liste des certificats approuvés par le client.
	TrustStore password	Saisissez le mot de passe utilisé pour vérifier l'intégrité des données TrustStore.
	Need Client authentication	Cochez cette case pour valider les données keystore. Vous devez ensuite compléter trois champs : - **KeyStore type** : sélectionnez le type de keystore à utiliser. Il peut être **PKCS 12** ou **JKS**. - **KeyStore file** : saisissez le chemin d'accès et le nom du fichier Keystore, ou parcourez votre répertoire jusqu'au fichier contenant les données keystore. - **KeyStore password** : saisissez le mot de passe keystore.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du Job ainsi qu'au niveau de chaque composant.
Utilisation	Ce composant est utilisé en standalone.	
Connections		Liens de sortie (d'un composant à un autre) : **Trigger** : Run if, On Subjob Ok, On Subjob Error, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Trigger** : Run if, On Subjob Ok, On Component Ok, On Component Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** Talend Open Studio.
Limitation	n/a.	

Scénario : Extraire des informations client d'un fichier WSDL privé

Ce scénario décrit un Job comprenant trois composants, qui se connecte à un fichier WSDL privé, pour extraire des informations client.

Le fichier WSDL utilisé dans ce Job accède au service Web correspondant via le protocole SSL. Le code correspondant dans le fichier est le suivant :

```
<wsdl:port name="CustomerServiceHttpSoap11Endpoint"
binding="ns:CustomerServiceSoap11Binding">
            <soap:address
location="https://192.168.0.22:8443/axis2/services/CustomerServic
e.CustomerServiceHttpSoap11Endpoint/"/>
        </wsdl:port>
```

Saisissez le code suivant dans le fichier *server.xml* de Tomcat :

```
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
               maxThreads="150" scheme="https" secure="true"
               clientAuth="true" sslProtocol="TLS"
     keystoreFile="D:/server.keystore" keystorePass="password"
     keystoreType="JKS"
     truststoreFile="D:/server.p12" truststorePass="password"
     truststoreType="PKCS12"
    />
```

Vous avez besoin de fichiers Keystore pour vous connecter à ce fichier WSDL. Pour reproduire ce Job, procédez comme suit :

- Déposez les composants suivants de la **Palette** dans l'espace de modélisation graphique : **tSetKeystore**, **tWebService**, et **tLogRow**.

- Cliquez-droit sur le composant **tSetKeystore** pour ouvrir son menu contextuel.
- Dans ce menu, sélectionnez **Trigger** > **On Subjob Ok** afin de relier ce composant au **tWebService**.
- Cliquez-droit sur le **tWebService** pour ouvrir son menu contextuel.
- Dans ce menu, sélectionnez **Row** > **Main** pour connecter ce composant au **tLogRow**.
- Double-cliquez sur le **tSetKeystore** pour ouvrir sa vue **Basic settings** et définir ses propriétés.

- Dans la liste **TrustStore type**, sélectionnez **PKCS12**.
- Dans le champ **TrustStore file**, parcourez votre répertoire jusqu'au fichier TrustStore. Ici, le fichier est *server.p12*.
- Dans le champ **TrustStore password**, saisissez le mot de passe pour ce fichier TrustStore. Dans cet exemple, le mot de passe est *password*.
- Cochez la case **Need Client authentication** afin d'activer les champs de configuration Keystore.
- Dans la liste **KeyStore type**, sélectionnez **JKS**.
- Dans le champ **KeyStore file**, parcourez votre répertoire jusqu'au fichier Keystore correspondant. Ici, le fichier est *server.keystore*.
- Double-cliquez sur le composant **tWebService** pour ouvrir l'éditeur du composant, ou sélectionnez le composant dans l'espace de modélisation graphique et dans sa vue **Basic settings**, cliquez sur le bouton **[...]** à côté du champ **Service configuration**.

- Dans le champ **WSDL**, parcourez votre répertoire jusqu'au fichier WSDL privé à utiliser. Dans cet exemple, ce fichier est *CustomerService.wsdl*.
- Cliquez sur le bouton **Refresh** à côté du champ **WSDL** afin de récupérer la description WSDL et l'afficher dans les champs qui suivent.
- Dans la liste **Port Name**, sélectionnez le port que vous souhaitez utiliser, *CustomerServiceHttpSoap11Endpoint* dans cet exemple.
- Dans la liste **Operation**, sélectionnez le service que vous souhaitez utiliser. Dans cet exemple, le service sélectionné est *getCustomer(parameters):Customer*.
- Cliquez sur **Next** pour ouvrir une nouvelle vue dans l'éditeur.

Dans le panneau de droite de la vue **Input mapping**, le paramètre d'entrée du service s'affiche automatiquement. Cependant, vous pouvez ajouter d'autres paramètres si vous sélectionnez **[+] parameters** et que vous cliquez sur le bouton **[+]** en haut, afin d'afficher la boîte de dialogue **[Parameter Tree]** dans laquelle vous pouvez sélectionner n'importe lequel des paramètres listés.

Le service Web de cet exemple n'a qu'un seul paramètre d'entrée, *ID*.

- Dans la colonne **Expression** de la ligne *parameters.ID*, saisissez entre guillemets l'ID du client qui vous intéresse. Dans cet exemple, l'ID est *A00001*.
- Cliquez sur **Next** pour ouvrir une nouvelle vue dans l'éditeur.

Dans la liste **Element** à gauche de la vue s'affiche automatiquement le paramètre de sortie du service Web. Cependant, vous pouvez ajouter d'autres paramètres si vous sélectionnez **[+] parameters** et que vous cliquez sur le bouton **[+]** en haut afin d'afficher la boîte de dialogue **[Parameter Tree]** dans laquelle vous pouvez sélectionner n'importe lequel des paramètres affichés.

Le service Web dans cet exemple contient quatre paramètres de sortie : *return.address*, *return.email*, *return.name* et *return.phone*.

Vous devez créer une connexion entre le paramètre de sortie du service Web défini et le schéma du composant de sortie. Pour ce faire :

- Dans le panneau à droite de la vue, cliquez sur le bouton **[...]** à côté du champ **Edit Schema** afin d'ouvrir une boîte de dialogue dans laquelle vous pouvez définir le schéma de sortie.

- Dans l'éditeur de schéma, cliquez sur le bouton **[+]** pour ajouter quatre colonnes au schéma de sortie.
- Cliquez dans chaque colonne et saisissez les nouveaux noms : *Name*, *Phone*, *Email* et *Address* dans cet exemple. Cela vous permettra de récupérer les informations client qui vous intéressent.

- Cliquez sur **OK** pour valider les modifications et fermer la boîte de dialogue du schéma.
- Dans la liste **Element** à droite de l'éditeur, glissez chaque paramètre dans le champ correspondant à la colonne que vous avez définie dans l'éditeur de schéma.

  S'il est disponible, vous pouvez utiliser le bouton **Auto map!**, situé en bas à gauche de l'interface, afin d'effectuer l'opération de mapping automatiquement.

- Cliquez sur **OK** pour valider vos modifications et femer l'éditeur.
- Dans l'espace de modélisation graphique, double-cliquez sur le composant **tLogRow** afin d'ouvrir sa vue **Basic settings** et définir ses propriétés.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent.
- Sauvegardez votre Job et appuyez sur **F6** pour l'exécuter.

Les informations du client dont l'ID est *A00001* sont retournées et affichées dans la console de **Talend Open Studio**.

```
-------------+--------------+-------------+-------------------------------------------
|                                   tLogRow_2
|=------------+--------------+-------------+-------------------+---------------------=
|name         |phone         |email        |address                                  |
|=------------+--------------+-------------+-------------------------------------------=
|Rose Gonzalez|(512) 757-9000|rose@edge.com|313 Constitution Place Austin, TX 78767 USA|
-------------+--------------+-------------+-------------------------------------------
[statistics] disconnected
Job tSetKeyStore ended at 19:01 24/11/2010. [exit code=0]
```

tSocketInput

Propriétés du tSocketInput

Famille de composant	Internet	
Fonction	Le composant **tSocketInput** ouvre un port de socket et permet d'écouter les données envoyées.	
Objectif	Le **tSocketInput** est un composant d'écoute, permettant de transmettre des données via un port défini.	
JAVA Basic settings	*Host name*	Nom ou adresse IP du serveur Hôte.
	Port	Port d'écoute à ouvrir.
	Timeout	Nombre de secondes avant fermeture du socket du port d'écoute.
	Uncompress	Cochez cette case pour extraire les données si nécessaire.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Row separator	Chaîne (ex : "\n"sous Unix) utilisée pour séparer les lignes.
	Escape Char	Caractère de la ligne à éviter.
	Text enclosure	Caractères utilisés pour entourer le texte.
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de Talend Open Studio.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de Talend Open Studio.

	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
Utilisation	Ce composant ouvre un point d'accès au poste de travail ou au serveur. Ce composant est un composant de début de Job et s'arrête uniquement après expiration.	
Limitation	n/a	

Les propriétés Perl étant légérement différentes des propriétés Java, elles sont décrites dans le tableau suivant :

PERL basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur Edit Schema pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Host name	Nom ou adresse IP du serveur Hôte
	Port	Port d'écoute à ouvrir.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	End of Line separator	Chaîne (ex : "\n" sous Unix) utilisée pour séparer les lignes.
	End of data	Caractère, chaîne ou expression régulière indiquant la fin de la section de données.
	Opening message /*Message* /*Acknowledge message* /*Closing message*	Description du message, si nécessaire.
Utilisation	Ce composant ouvre un point d'accès à un poste de travail ou à un serveur. Ce composant débute un Job et ne s'arrête qu'après avoir reçu un message de fermeture.	

Scénario : Transférer des données vers un port d'écoute (Java)

Le scénario suivant décrit un double Job dont le but est de transférer des données via un port d'écoute. Le composant Socket peut aussi permettre de contrôler la communication entre les serveurs ne pouvant pas communiquer directement entre eux.

- Créez deux Jobs : le premier (*SocketInput*) ouvre un port d'écoute et attend que les données soient envoyées. Le deuxième (*SocketOutput*) transmet des données délimités d'un fichier vers le numéro de port correspondant au port d'écoute.
- Dans le premier Job, cliquez et déposez les composants suivants : **tSocketInput** et **tLogRow**.
- Dans le deuxième Job, cliquez et déposez les composants suivants : **tFileInputDelimited** et **tSocketOutput**.
- Commencez par configurer les paramètres du deuxième Job.
- Sélectionnez **tFileInputDelimited** et dans l'onglet **Basic Settings** de la vue **Component**, configurez les paramètres d'accès au fichier d'entrée.

- Dans le champ **File Name**, renseignez le chemin d'accès au fichier.
- Renseignez les séparateurs de lignes et de champs respectivement dans les champs **Row** et **Field separators**, ainsi que l'en-tête dans le champ **Header**.

- Décrivez le schéma des données à passer au composant **tSocketOutput**.
- Sélectionnez le composant **tSocketOutput** et configurez les paramètres de l'onglet **Basic Settings** de la vue **Component**.

- Dans le champ **Host**, renseignez l'adresse IP et dans le champ **Port**, renseignez le numéro du port vers lequel les données seront transférées.
- Dans le champ **Retry**, définissez le nombre d'essais à effectuer et dans le champ **Timeout** renseignez la durée (en seconde) avant que le Job ne s'arrête automatiquement.
- Configurez le reste des options si nécessaire.
- Le schéma devrait se propager du composant précédent.
- Maintenant, configurez les paramètres du premier Job (*SocketInput*) contenant le composant **tSocketInput**.

- Dans le champ **Host**, renseignez l'adresse IP et dans le champ **Port**, renseignez le numéro du port d'écoute vers lequel les données sont transférées.
- Dans le champ **Timeout**, définissez la durée (en secondes) avant que le Job ne s'arrête automatiquement.

- Configurez le reste des options si nécessaire.
- Modifiez le schéma afin qu'il corresponde en tout point ou en partie au schéma du deuxième Job.
- Laissez les propriétés par défaut du composant **tLogRow**.
- Exécutez le Job *SocketInput* en premier, afin d'ouvrir le port d'écoute et de le préparer à recevoir les données.
- Avant que le Job ne s'arrête, lancez l'autre Job (SocketOutput) afin de transmettre les données.

Le résultat s'affiche dans la vue **Run**, ainsi que les informations d'ouverture du socket.

```
Starting job SocketInput at 17:53 04/02/2008.
socket connected
AL|Alabama|Montgomery|Birmingham
AK|Alaska|Juneau|Anchorage
AZ|Arizona|Phoenix|Phoenix
AR|Arkansas|Little Rock|Little Rock
CA|California|Sacramento|Los Angeles
CO|Colorado|Denver|Denver
CT|Connecticut|Hartford|Bridgeport
DE|Delaware|Dover|Wilmington
FL|Florida|Tallahassee|Jacksonville
GA|Georgia|Atlanta|Atlanta
HI|Hawaii|Honolulu|Honolulu
ID|Idaho|Boise|Boise
IL|Illinois|Springfield|Chicago
IN|Indiana|Indianapolis|Indianapolis
```

tSocketOutput

Propriétés du tSocketOutput

Famille de composant	Internet	
Fonction	Le composant **tSocketOutput** écrit des données sur un port d'écoute.	
Objectif	Le **tSocketOutput** transfert des données d'un flux d'entrée vers le port d'écoute du socket.	
Basic settings	*Host*	Nom ou adresse IP du serveur Hôte.
	Port	Port d'écoute à ouvrir
	Compress	Cochez cette case pour archiver les données si nécessaire.
	Retry times	Nombre d'essais avant que le Job ne soit mis en échec.
	Timeout	Nombre de secondes avant fermeture du port d'écoute.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	Row separator	Chaîne (ex : "\n" sous Unix) séparant les lignes.
	Escape Char	Caractère de la ligne à éviter.
	Text enclosure	Caractéres utilisés pour entourer le texte.
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur Edit Schema pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du Guide utilisateur de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du Guide utilisateur de **Talend Open Studio**.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
Utilisation	Ce composant ouvre un point d'accès au poste de travail ou au serveur. Ce composant est un composant de début de Job et s'arrête uniquement après expiration.	

Limitation	n/a

Scénario associé

Pour un exemple d'utilisation du **tSocketOutput**, consultez *Scénario : Transférer des données vers un port d'écoute (Java), page 1116.*

tSOAP

Propriétés du tSOAP

Famille de composant	Internet	
Fonction	Le composant **tSOAP** envoie le message SOAP défini avec les paramètres donnés au Service Web invoqué et retourne la valeur comme définie, à partir des paramètres donnés.	
Objectif	Ce composant appelle une méthode via un Service Web afin de récupérer les valeurs des paramètres définiese dans l'éditeur du composant.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour apporter des modifications au schéma. Notez que si vous effectuez des modifications, le schéma devient automatiquement built-in. Cliquez sur **Sync columns** afin de récupérer le schéma du composant précédent dans le Job. Le schéma de ce composant est en lecture seule, et contient trois colonnes : *Header*, *Body*, *Fault*.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Use NTLM	Cochez cette case si vous utilisez un protocole d'authentification NTLM. **Domain** : Nom de domaine du client.
	Need authentication	Cochez la case d'authentification et renseignez le nom de l'utilisateur dans le champ **Username** et son mot de passe dans le champ **Password**, si cela est nécessaire pour accéder au service.
	Use http proxy	Cochez cette case si vous vous connectez derrière un proxy et renseignez les informations correspondantes.

	Trust server with SSL	Cochez cette case pour authentifier le serveur auprès du client via un protocole SSL et renseignez les champs correspondants. **TrustStore file** : saisissez le chemin d'accès et le nom du fichier TrustStore contenant la liste des certificats approuvés par le client. **TrustStore password** : saisissez le mot de passe utilisé pour vérifier l'intégrité des données TrustStore.
	ENDPOINT	Saisissez l'URL du serveur Web invoqué.
	SOAP action	Saisissez l'URL de l'en-tête HTTP SOAPAction à utiliser pour identifier le but de la requête HTTP SOAP.
	SOAP version	Sélectionnez la version du système SOAP que vous utilisez. *L'enveloppe SOAP requise varie selon les versions.*
	SOAP message	Saisissez le message SOAP à envoyer au Service Web invoqué. Les variables globales et de contexte peuvent être utilisées lorsque vous écrivez un message SOAP. Pour plus d'informations concernant les variables de contexte, consultez *Variables, page 256* du **Guide Utilisateur Talend Open Studio**.
Advanced settings	*Temporary folder (for wsdl2java)*	Définissez (ou parcourez votre répertoire jusqu'à) un dossier temporaire configuré où stocker les fichiers WSDL.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en tant que composant d'entrée ou intermédiaire.	
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : Run if, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Iterate **Trigger** : Run if, On Component Ok, On Component Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur Talend Open Studio**.
Limitation	N/A	

Scénario : Extraire des informations météo en utilisant un Service Web

Ce scénario Java décrit un Job à deux composants utilisant un Service Web pour récupérer les informations concernant la météo d'une ville américaine.

Le service Web à utiliser est ; http://www.deeptraining.com/webservices/weather.asmx.

- Déposez les composants suivants de la **Palette** dans l'espace de modélisation graphique : **tSOAP** et **tLogRow**.

- Cliquez-droit sur le composant **tSOAP** afin d'ouvrir le menu contextuel.
- Sélectionnez **Row** > **Main**.
- Cliquez sur le **tLogRow** pour relier les composants à l'aide d'un lien **Main Row**.
- Double-cliquez sur le **tSOAP** afin d'ouvrir sa vue **Basic settings** et définir ses propriétés.

- Dans le champ **ENDPOINT**, saisissez ou collez l'URL du Service Web à utiliser, entre guillemets : "*http://www.deeptraining.com/webservices/weather.asmx*".
- Dans le champ **SOAP Action**, saisissez ou collez l'URL de l'en-tête HTTP SOAPAction permettant de déterminer que vous souhaitez récupérer les informations concernant la météo : *http://litwinconsulting.com/webservices/GetWeather*.

Vous pouvez voir cette adresse en regardant le WSDL du Service Web que vous appelez. Pour le Service Web de cet exemple, dans votre navigateur, saiaissez `?wsdl` à la fin de l'URL du Service Web utilisé dans le champ **ENDPOINT**, ouvrez la page Web correspondante, puis regardez l'action SOAPAction définie sous le noeud Operation :

```
<wsdl:operation name="GetWeather">
<soap:operation
soapAction="http://litwinconsulting.com/webservices/GetWeathe
r" style="document"/>
```

- Dans le champ **SOAP version**, sélectionnez la version du système SOAP utilisé. Dans ce scénario, la version est *SOAP 1.1*.
- Dans le champ **SOAP message**, saisissez le message, au format XML, utilisé pour récupérer les informations du Service Web invoqué. Dans cet exemple, récupérez les informations météo de Chicago, saisissez donc :

```
"<soapenv:Envelope
xmlns:soapenv=\"http://schemas.xmlsoap.org/soap/envelope/\"
xmlns:web=\"http://litwinconsulting.com/webservices/\">
 <soapenv:Header/>
  <soapenv:Body>
     <web:GetWeather>
         <web:City>Chicago</web:City>
      </web:GetWeather>
   </soapenv:Body>
 </soapenv:Envelope>"
```

- Sauvegardez votre Job et appuyez sur **F6** pour l'exécuter.

La météo de Chicago est récupérée et affichée dans la console de la vue **Run**.

tWebServiceInput

Propriétés du tWebServiceInput

Famille de composant	Internet	
Fonction	Ce composant appelle la méthode du service Web invoqué et retourne la classe définie en fonction des paramètres donnés.	
Objectif	Le **tWebServiceInput** invoque une méthode via un service Web dont l'objectif est celui décrit dans les propriétés. Pour traiter des données complexes hiérarchiques, vous devez utiliser les fonctions avancées du **tWebServiceInput** en codant directement du Java dans le champ **Code** de la vue **Advanced settings**.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier dans lequel sont stockées les propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
		Cliquez sur cette icône pour ouvrir l'assistant de création de schéma WSDL et enregistrer les paramètres de connexion WSDL que vous avez définis dans la vue **Basic settings** du composant dans le Repository. Pour plus d'informations sur la configuration et le stockage des paramètres de connexion WSDL, consultez la section *Configurer un schéma Web Service* du **Guide Utilisateur** de **Talend Open Studio**.
Perl uniquement	*Encoding*	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données des bases de données.
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champ qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du Guide Utilisateur de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du Guide utilisateur de **Talend Open Studio**.

Perl uniquement	*End Point URI*	Identificateur de ressources du service Web.
	WSDL	Description des bindings et de la configuration du service Web.
Java uniquement	*Need authentication / Username et Password*	Cochez cette case d'authentification et : - Saisissez un identifiant, ainsi qu'un mot de passe, dans les champs correspondants, si nécessaire pour accéder au service. Sinon, - Cochez la case **Windows authentication** et saisissez le nom de domaine Windows dans le champ correspondant si cela est nécessaire pour accéder au service.
Java uniquement	*Use http proxy*	Cochez cette case si vous vous connectez derrière un proxy et renseignez les informations correspondantes.
Java uniquement	*Trust server with SSL*	Cochez cette case pour authentifier le serveur auprès du client via un protocole SSL et renseignez les champs correspondants. **TrustStore file** : saisissez le chemin d'accès et le nom du fichier TrustStore contenant la liste des certificats approuvés par le client. **TrustStore password** : saisissez le mot de passe utilisé pour vérifier l'intégrité des données TrustStore.
Java uniquement	*Time out (second)*	Saisissez en secondes la durée de connexion au service Web.
	Method Name	Saisissez le nom exact de la méthode à invoquer. Le nom de la méthode DOIT correspondre à la méthode décrite dans le service Web. Le nom de la méthode est sensible à la casse.
	Parameters	Saisissez les paramètres attendus et les valeurs à retourner. Assurez-vous que les paramètres saisis correspondent parfaitement aux noms et à la casse des paramètres décrits dans la méthode.
Advanced settings	*Advanced Use* *Java uniquement*	Cochez cette case pour afficher les champs suivants permettant d'utiliser les fonctions avancées du **tWebServiceInput** : **WSDL2Java** : cliquez sur le bouton **[...]** pour générer des routines contenant les codes Java nécessaires à la connexion et à l'interrogation du service Web. **Code** : saisissez le code permettant de se connecter et d'interroger le service Web en vous basant sur les deux modèles proposés en utilisant les routines générées automatiquement. **Match Brackets** : sélectionnez le nombre d'accolades à utiliser pour fermer la boucle `for` en fonction du nombre d'accolades ouvertes.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce component est généralement utilisé comme composant de début. Il requiert un composant de sortie.
Limitation	n/a

Scénario 1 : Extraire des images à l'aide d'un Webservice

Ce scénario décrit un Job constitué de deux composants dont le but est d'utiliser une méthode de service Web et d'afficher le résultat en sortie dans une vue standard.

La méthode récupère une adresse URL complète sous forme de chaîne en entrée et retourne une liste d'images provenant d'une page Web donnée.

- Cliquez et déposez les composants **tWebServiceInput** et **tLogRow**.
- Dans la vue **Component** du composant **tWebServiceInput**, configurez les spécifications WSDL : **WSDL** et **End Point URI** lorsque cela est nécessaire.
- Si le service Web invoqué requiert des informations d'authentification, cochez la case **Need authentication?** et renseignez ces informations.

- Si vous vous connectez derrière un proxy, cochez la case **Use http proxy** et renseignez les champs concernant l'hôte, l'utilisateur, le port et le mot de passe.
- Dans le champ **Method Name**, saisissez le nom de la méthode tel que défini dans la description du service Web. Le nom et la casse de la méthode saisie doivent correspondre parfaitement à la méthode du service Web demandé.

- Dans la zone **Parameters**, cliquez sur le bouton **[+]** pour ajouter une ligne au tableau, puis saisissez le nom exact des paramètres tel qu'attendus par la méthode.
- Dans la colonne **Value**, saisissez l'adresse URL du site Web à partir duquel les images seront extraites.
- Connectez le composant **tWebServiceInput** au composant de sortie standard, **tLogRow**.
- Appuyez sur **F6** pour exécuter le Job.

```
Starting job WebService at 16:30 06/06/2007.
<img src="http://us.a1.yimg.com/us.yimg.com/i/ww/beta/y3.gif"></img>
<img src="http://us.i1.yimg.com/us.yimg.com/i/ww/tbl/webs.gif"></img>
<img src="http://us.i1.yimg.com/us.yimg.com/i/ww/tbl/fdbg.gif"></img>
<img src="http://us.i1.yimg.com/us.yimg.com/i/ww/tbl/fdbg.gif"></img>
<img src="http://us.i1.yimg.com/us.yimg.com/i/ww/tbl/fdbg.gif"></img>
<img src="http://us.i1.yimg.com/us.yimg.com/i/ww/tbl/fdbg.gif"></img>
<img src="http://us.i1.yimg.com/us.yimg.com/i/ww/tbl/fdbg.gif"></img>
<img src="http://us.i1.yimg.com/us.yimg.com/i/ww/tbl/fdbg.gif"></img>
<img src="http://us.i1.yimg.com/us.yimg.com/i/ww/tbl/allys.gif"></img>
Job WebService ended at 16:30 06/06/2007. [exit code=0]
```

Toutes les images extraites du site Web spécifié sont retournées dans la vue **Run**, sous la forme d'une liste d'adresses URL.

Scénario 2 : Lire des données publiées dans un service Web, en utilisant les fonctions avancées du tWebServiceInput (Java)

Ce scénario décrit un Job constitué de deux composants dont le but est de récupérer une liste de fonds publiée sur un service Web financier (distribué par www.xignite.com) et d'afficher le résultat en sortie dans la vue standard (la vue **Run**).

Ce scénario est destiné à des utilisateurs avancés ayant des notions de Java. En effet, l'objectif de ce Job étant de récupérer des données complexes hiérarchiques, il est nécessaire de coder en Java les fonctions dont vous avez besoin pour récupérer ces données.

- Cliquez et déposez le composant **tWebServiceInput** de la famille **Internet** et le composant **tLogRow** de la famille **Logs & Errors** dans l'espace de modélisation.
- Reliez les composants à l'aide d'une connexion de type **Row Main**.
- Double-cliquez sur le **tWebServiceInput** pour afficher la vue **Component** et paramétrer ses propriétés :

Dans l'onglet **Basic settings** :

- Dans la liste **Property Type**, sélectionnez l'option **Built-in** et renseignez manuellement les champs suivants.
- Dans la liste **Schema Type**, sélectionnez **Built-in** et cliquez sur le bouton **[...]** pour configurer manuellement le schéma comme dans la figure ci-dessous :

- Cliquez sur **OK** pour valider le schéma et fermer la fenêtre.
- Une boîte de dialogue s'ouvre et vous demande si vous souhaitez propager les modifications. Cliquez sur **Yes** (Oui).
- Dans le champ **WSDL**, saisissez l'adresse du WSDL auquel vous connecter.
- Dans le champ **Time out**, saisissez en secondes la durée de la connexion au service Web.
- Cliquez sur l'onglet **Advanced settings** pour renseigner les fonctions avancées du composant **tWebServiceInput** :

- Cochez la case **Advanced Use** pour afficher les champs de configuration avancée.
- Cliquez sur le bouton **[...]** situé à côté du champ **WSDL2Java** pour générer automatiquement des routines à partir du WSDL du service Web.

Ces routines apparaissent sous le noeud **Code** > **Routines** du **Repository**, et peuvent être utilisées dans le code pour construire les fonctions nécessaires à la récupération des données hiérarchiques du service Web.

- Saisissez les fonctions adéquates dans le champ **Code**. Par défaut, deux exemples de code sont fournis dans le champ **Code** : le premier sert à retourner une donnée et le deuxième à en retourner plusieurs.
- Pour ce scénario, l'objectif est de retourner plusieurs données. Ainsi, supprimez le premier exemple de code et utilisez le deuxième exemple de code pour construire les fonctions.
- Dans le deuxième exemple, remplacez les morceaux de code exemple par le code correspondant généré dans les routines à partir du WDSL.
- Remplacez `TalendJob_PortType` par le nom de la routine terminant par **_Port_Type** : `XigniteFundHoldingsSoap_PortType`.
- Remplacez les différentes instances de `TalendJob` par un nom plus parlant tel que le nom de la méthode utilisée : *fundHolding*.
- Remplacez `TalendJobServiceLocator` par le nom de la routine terminant par **Locator** : `XigniteFundHoldingsLocator`.
- Remplacez les deux instances de `TalendJobSoapBindingStub` par le nom de la routine terminant par **BindingStub** : `XigniteFundHoldingsSoap_BindingStub`.
- Dans les parenthèses correspondant aux codes `stub.setUsername` et `stub.setPassword`, saisissez respectivement votre identifiant et votre mot de passe entre guillemets.
  Dans un souci de confidentialité ou de maintenance, vous pouvez mettre votre identifiant et votre mot de passe dans des variables de contexte.

- La liste des fonds fournis par le service Web de Xignite sont identifiés à l'aide de "symboles" (**Symbol**), qui sont de type **String** (chaîne de caractères). Dans cet exemple, l'objectif est de récupérer la liste des fonds dont la première lettre du symbole est comprise entre "I" et "J". Pour cela, définissez les arguments à utiliser : `string startSymbol="I"` et `String endSymbol="J"`.
- Puis saisissez le code permettant de créer le tableau de résultat récupérant la liste des fonds (**listFunds**) de participation (**fundholdings**) en utilisant les arguments définis : `routines.Fund[] result = fundHoldings.listFunds(startSymbol, endSymbol);`
- Effectuez une boucle sur la liste des fonds pour récupérer les fonds compris entre "I" et "J" : `for(int i = 0; i < result.length; i++) {`
- Puis définissez les sorties à retourner, par exemple : récupérez les données **CIK** du schéma **Security** avec le code `getSecurity().getCIK()` et transmettez-les au champ **CIK** du schéma de sortie.

Le code obtenu pour faire fonctionner le service Web doit correspondre au code suivant :

```
routines.XigniteFundHoldingsSoap_PortType fundHoldings = new
routines.XigniteFundHoldingsLocator().getXigniteFundHoldingsSoap(
);

routines.XigniteFundHoldingsSoap_BindingStub stub =
(routines.XigniteFundHoldingsSoap_BindingStub)fundHoldings;

stub.setUsername("identifiant");
stub.setPassword("mot de passe");

String startSymbol="I";
String endSymbol="J";

routines.Fund[] result = fundHoldings.listFunds(startSymbol,
endSymbol);

for(int i = 0; i < result.length; i++) {

 output_row.CIK = (result[i]).getSecurity().getCIK();
 output_row.cusip = (result[i]).getSecurity().getCusip();
 output_row.symbol = (result[i]).getSecurity().getSymbol();
 output_row.ISIN = (result[i]).getSecurity().getISIN();
 output_row.valoren = (result[i]).getSecurity().getValoren();
 output_row.name = (result[i]).getSecurity().getName();
 output_row.market = (result[i]).getSecurity().getMarket();
 output_row.category =
(result[i]).getSecurity().getCategoryOrIndustry();
 output_row.asOfDate = (result[i]).getAsOfDate();
```

Les sorties définies dans le code Java `output_row.sortie` doivent correspondre exactement aux noms des colonnes définis dans le schéma du composant. En effet, la casse doit être respectée pour que les données soient récupérées.

- Dans le champ **Match Brackets**, sélectionnez le nombre d'accolades à utiliser pour fermer la boucle `for` en fonction du nombre d'accolades ouvertes. Pour ce scénario, sélectionnez une seule accolade puisqu'une seule accolade a été ouverte dans le code.
- Double-cliquez sur le composant **tLogRow** pour afficher la vue **Component** et paramétrer ses propriétés.
- Cliquez sur le bouton **[...]** situé à côté du champ **Edit Schema** pour vérifier que le schéma du composant précédent a bien été propagé au composant de sortie. Sinon, cliquez sur le bouton **Sync columns** pour récupérer le schéma.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job tWebServiceInput_scenario2 at 13:59 27/07/2009.

[statistics] connecting to socket on port 3922
[statistics] connected

|893957753|IAAAX|||Ta Idex Asset Allocation Growth |FUNDS|Large
Blend|7/27/2009
|893957746|IAABX|||Ta Idex Asset Allocation Growth |FUNDS|Large
Blend|7/27/2009
|893957720|IAALX|||Ta Idex Asset Allocation Growth |FUNDS|Large
Blend|7/27/2009
0000895430|44980Q724|IACAX|||ING International Capital Appre
|FUNDS|Foreign Large Growth|7/27/2009
N/A|893958454|IACBX|||Ta Idex Salomon All Cap Class B |FUNDS|Large
Blend|7/27/2009
N/A|00141T122|IACFX|||Aim China Institutional Class
|FUNDS|Pacific/Asia ex-Japan Stk|7/27/2009
```

Les fonds compris entre "I" et "J" sont retournés et affichés dans la console de **Talend Open Studio**.

tXMLRPCInput

Propriétés du tXMLRPCInput

Famille de composant	Internet	
Fonction	Ce composant appelle la méthode du service RPC invoqué et retourne la classe définie en fonction des paramètres donnés.	
Objectif	Le composant **tXMLRPCInput** invoque une méthode via un service Web dont l'objectif est celui décrit dans les propriétés.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent. Dans le contexte du RPC, le schéma correspond aux paramètres de sortie. Si deux paramètres doivent être retournés, alors le schéma comportera deux colonnes.
	Server URL	Adresse URL du service RPC à accéder
	Need authentication / *Username et Password*	Cochez la case d'authentification et renseignez le nom de l'utilisateur (username) et son mot de passe (password), si cela est nécessaire pour accéder au service.
	Method Name	Saisissez le nom exact de la méthode à invoquer. Le nom de la méthode DOIT correspondre parfaitement à la méthode décrite dans le service RPC. Le nom de la méthode est sensible à la casse.
	Return class	Sélectionnez le type de données à retourner par la méthode. Assurez-vous qu'il correspond parfaitement à celui défini dans la méthode.
	Parameters	Saisissez les paramètres attendus en entrée par la méthode.
Utilisation	Ce composant est généralement utilisé comme composant de début. Il nécessite un composant de sortie.	
Limitation	n/a	

Scénario : Chercher le nom d'un Etat via une méthode XMLRPC

Ce scénario décrit un Job à deux composants utilisant une méthode RPC et affichant le résultat dans la console.

- Cliquez et déposez les composants **tXMLRPCInput** et **tLogRow**.
- Dans l'onglet **Basic settings**, paramétrez les propriétés du composant **tXMLRPCInput**.

- Dans la liste déroulante **Schema type**, sélectionnez l'option **Built-in**.
- Configurez un schéma contenant une unique colonne puisque seul un paramètre est attendu en sortie : *StateName*.

- Dans le champ **Server url**, renseignez l'adresse URL du serveur. Pour cet exemple, utilisez : *http://phpxmlrpc.sourceforge.net/server.php*
- Aucune information d'authentification n'est nécessaire pour cet exemple.
- Dans le champ **Method**, renseignez la méhode à appeler, ici : *examples.getStateName*
- Pour cet exemple, la classe retournée n'est pas obligatoire. Ainsi, dans le champ **return class**, laissez les paramètres par défaut.
- Puis, dans le tableau **Parameters**, configurez les paramètres nécessaires à la méthode appelée. La colonne **Name** n'est pas utilisée dans le code mais la valeur doit correspondre à la syntaxe attendue par la méthode. Dans cet exemple, saisissez *State Nr* dans le champ **Name** et dans le champ **Value**, choisissez un numéro au hasard, ici : *42*.

- La classe n'a pas beaucoup d'impact sur cette méthode démo mais pourrait en avoir sur une autre méthode. Ainsi, laissez les paramètres par défaut.
- Dans la vue **Component** du composant **tLogRow**, cochez la case **Print schema column name in front of each value**.
- Enregistrez le Job puis exécutez-le, en appuyant sur **F6**.

```
Starting job xmlrpc at 16:25 21/09/2007.
StateName: South Dakota
Job xmlrpc ended at 16:25 21/09/2007. [exit code=0]
```

Le résultat obtenu en sortie en utilisant la méthode RPC `GetStateName` et correspondant au quarante deuxième Etat des Etats Unis est donc *South Dakota*.

Composants Logs & Errors

Ce chapitre passe en revue les principaux composants de la famille **Logs & Errors** de la Palette de **Talend Open Studio**.

La famille **Logs & Errors** regroupe les composants dédiés à la gestion des données de log et des erreurs des Jobs.

tAssert

Propriétés du tAssert

Le composant **tAssert** est étroitement lié au **tAssertCatcher** pour tester une exécution de Job. Il génère une assertion booléenne à partir d'une instruction conditionnelle, dans le but d'alimenter le statut d'exécution présenté par le **tAssertCatcher**.

Famille de composant	Logs & Errors	
Fonction	Envoie le message de statut du Job au **tAssertCatcher**.	
Objectif	Génère l'assertion booléenne relative à l'analyse du Job. Le statut peut être : **Ok** : l'exécution du Job a réussi. **Fail** : l'exécution a échoué. Les résultats du Job testé ne correspondent pas à ce qui était attendu, ou une erreur est survenue lors de l'exécution.	
Basic settings	*Description*	Saisissez votre message descriptif afin d'identifier l'assertion du **tAssert**.
	Expression	Saisissez l'instruction conditionnelle sur laquelle baser votre assertion.
Utilisation	Ce composant suit l'action à laquelle est liée directement la condition assertive. Il peut être utilisé comme composant intermédiaire ou final du Job principal, ou bien en tant que composant de début, intermédiaire ou de fin du Job secondaire.	
Limitation	L'évaluation du **tAssert** est capturée par le **tAssertCatcher**.	

Scénario : Paramétrer une condition assertive pour l'exécution d'un Job

Ce scénario décrit comment paramétrer une condition assertive dans le composant **tAssert** afin d'évaluer si l'exécution du Job a réussi ou non. De plus, vous pouvez également trouver comment deux résultats différents d'évaluation s'affichent et comment les lire. Le scénario utilise un **tAssert**, mais aussi :

- un **tFileInputDelimited** et un **tFileOutputDelimited**. Les deux composants constituent le Job principal, duquel le statut d'exécution est évalué. Pour plus d'informations sur ces deux composants, consultez les composants : *tFileInputDelimited, page 915* et *tFileOutputDelimited, page 978*.
- un **tFileCompare**. Il réalise la comparaison entre le fichier de sortie du Job principal et un fichier de référence standard. Le résultat comparatif est évalué par le **tAssert** par rapport à la condition assertive configurée dans ses paramètres. Pour plus d'informations à propos du tFileCompare, consultez le composant *tFileCompare, page 896*.
- un **tAssertCatcher**. Il capture l'évaluation générée par le **tAssert**. Pour plus d'informations à propos du composant **tAssertCatcher**, consultez le composant *tAssertCatcher, page 1144*.

- un **tLogRow**. Il vous permet de lire l'évaluation capturée. Pour plus d'informations à propos du **tLogRow**, consultez le composant *tLogRow, page 1163*.

Afin de créer le Job principal, procédez comme suit :

- Préparez un fichier délimité *.csv* comme fichier source lu par votre Job principal.
- Editez deux lignes dans ce fichier. Les modifications n'ont pas d'importance, vous pouvez uniquement simplifier le contenu.
- Nommez-le *source.csv*.
- Dans **Talend Open Studio**, créez un nouveau Job, que vous appelerez *JobAssertion*.
- Placez un **tFileInputDelimited** et un **tFileOutputDelimited** dans l'espace de modélisation graphique.
- Connectez- les à l'aide d'un lien **Row Main** afin de créer le Job principal.

- Double-cliquez sur le **tFileInputDelimited** afin d'ouvrir sa vue **Component**.
- Dans le champ **File Name** de la vue **Component**, renseignez le chemin d'accès à votre fichier *source.csv*.

- Sélectionnez dans la liste **Property Type** le mode **Built-In** et cliquez sur le bouton **[...]** de l'option **Edit schema** afin de définir les données à passer au **tFileOutputDelimited**. Dans ce scénario, sélectionnez les données présentées dans le fichier *source.csv* que vous avez créé.

Pour plus d'informations à propos des types de schéma, consultez *Paramétrer un schéma built-in* et *Paramétrer un schéma du Repository* du **Guide Utilisateur Talend Open Studio**.

- Définissez les autres paramètres dans les champs correspondants, selon le fichier *source.csv* que vous avez créé.
- Double-cliquez sur le **tFileOutputDelimited** afin d'ouvrir sa vue **Component**.

- Dans le champ **File Name** de la vue **Component**, renseignez le chemin d'accès à votre fichier de sortie, et laissez les autres champs comme définis par défaut.

- Appuyez sur **F6** afin d'exécuter le Job principal. Il lit votre fichier *source.csv*, passe les données au **tFileOutputDelimited** et écrit un fichier délimité de sortie, *out.csv*.

Continuez à modifier votre Job, afin de voir comment le **tAssert** évalue le statut d'exécution du Job principal.

- Renommez le fichier *out.csv* en *reference.csv*. Ce fichier est le résultat attendu de l'écriture du Job principal.
- Placez un **tFileCompare**, un **tAssert** et un **tLogRow** dans l'espace de modélisation graphique.
- Connectez-les avec un lien **Row Main**.
- Connectez le **tFileInputDelimited** au **tFileCompare** avec un lien **OnSubjobOk**.

- Double-cliquez sur le **tFileCompare** afin d'ouvrir sa vue **Component**.
- Dans la vue **Component**, renseignez les chemins d'accès dans les champs **File to compare** et **Reference file** correspondants, et laissez les autres champs comme définis par défaut.

Pour plus d'informations à propos du composant **tFileCompare**, consultez le composant *tFileCompare, page 896*.

- Cliquez ensuite sur le **tAssert**, puis cliquez sur l'onglet **Component**, au bas de l'espace de modélisation graphique.

- Dans la vue **Component**, éditez la condition `row2.differ==0`, dans le champ **expression** ainsi que la **description** de l'assertion.

Dans le champ **expression**, `row2` constitue le flux de données transmis du **tFileCompare** au **tAssert**, `differ` est une colonne du schéma du **tFileCompare** qui affiche si les fichiers comparés sont identiques. 0 signifie qu'aucune différence n'a été détectée entre le fichier *out.csv* et le fichier *reference.csv* par le **tFileCompare**. Les fichiers comparés sont donc identiques, la condition assertive est remplie, le **tAssert** conclut que le Job principal a été exécuté avec succès. Sinon, il conclut qu'il a échoué.

La colonne *differ* fait partie du schéma, en lecture seule, du **tFileCompare**. Pour plus d'informations sur son schéma, consultez le composant *tFileCompare, page 896*.

- Appuyez sur **F6** afin d'exécuter votre Job.
- Vérifiez le résultat retourné dans la vue **Run**.

```
Starting job JobAssertion at 14:00 29/01/2010.

[statistics] connecting to socket on port 3788
[statistics] connected
[job JobAssertion] Files are identical
.----------+----------------+-------------------+------------+--------------+------+---------
-----------------------------.
|                                                        tLogRow_1
|
|=---------+----------------+-------------------+------------+--------------+------+---------
-----------------------------=|
|file      |file_ref        |moment             |job         |component     |differ|message
|
|=---------+----------------+-------------------+------------+--------------+------+---------
-----------------------------=|
|C:/out.csv|C:/reference.csv|2010-01-29 14:00:17|JobAssertion|tFileCompare_1|0     |[job
JobAssertion] Files are identical|
'----------+----------------+-------------------+------------+--------------+------+---------
-----------------------------'
[statistics] disconnected
Job JobAssertion ended at 14:00 29/01/2010. [exit code=0]
```

La console affiche les résultats de la comparaison du **tFileCompare** : `Files are identical`. Mais l'évaluation du **tAssert** n'est visible nulle part.

Vous avez donc besoin du composant **tAssertCatcher** pour capturer l'évaluation.

- Placez un **tAssertCatcher** et un **tLogRow** dans l'espace de modélisation graphique.
- Connectez-les à l'aide d'un lien **Row Main**.

- Utilisez la configuration par défaut du **tAssertCatcher** dans sa vue **Component**.

- Appuyez sur **F6** afin d'exécuter votre Job.
- Vérifiez les résultats présentés dans la vue **Run**. Vous pouvez voir que les informations sur le statut du Job ont été ajoutées :
  `2010-01-29 15:37:33|fAvAzH|TASSERT|JobAssertion|java|tAssert_1|Ok|--|The output file should be identical with the reference file.`

```
Starting job JobAssertion at 19:33 01/02/2010.

[statistics] connecting to socket on port 3556
[statistics] connected
[job JobAssertion] Files are identical
2010-02-01
19:33:40|r9FqSQ|TASSERT|JobAssertion|java|tAssert_1|Ok|--|The
output file should be identical with the reference file.
```

La description du *JobAssertion* dans la console est organisée selon le schéma du **tAssertCatcher**, lequel comprend des colonnes correspondant au temps d'exécution, à l'ID traité, au nom du projet, au langage du code, l'origine de l'évaluation, le résultat de l'évaluation, les informations détaillées de l'évaluation, et le message descriptif de l'assertion. Pour plus d'informations à propos du schéma du **tAssertCatcher**, consultez le composant *tAssertCatcher, page 1144*.

La console indique que le statut d'exécution du Job *JobAssertion* est *Ok*. En plus de l'évaluation, vous pouvez toujours voir les informations descriptives du *JobAssertion*, parmi lesquelles la description que vous avez saisie dans les **Basic settings** du **tAssert**.

Effectuez ensuite quelques opérations pour que le Job principal ne puisse générer le fichier attendu. Pour ce faire, procédez comme suit :

- Effacez une ligne de votre fichier *reference.csv*.
- Appuyez sur **F6** pour exécuter votre Job à nouveau.
- Vérifiez les résultats retournés dans la vue **Run**.
  `2010-02-01 19:47:43|GeHJNO|TASSERT|JobAssertion|tAssert_1|Failed|Test logically failed|The output file should be identical with the reference file.`

```
Starting job JobAssertion at 19:47 01/02/2010.

[statistics] connecting to socket on port 4001
[statistics] connected
[job JobAssertion] Files differ
2010-02-01
19:47:43|GeHJNO|TASSERT|JobAssertion|java|tAssert_1|Failed|Test
logically failed|The output file should be identical with the
reference file.
```

La console indique que le statut d'exécution du Job principal est *Failed* (a échoué). L'explication détaillée suit peu après, *Test logically failed* (échec logique du test).

Vous pouvez avoir une idée du statut de votre Job : il a échoué lors de la génération du fichier attendu à cause d'un échec logique. Cet échec logique peut provenird'une erreur logique lors de la création du Job.

Le statut et ses explications sont présentés respectivement dans les colonnes de statut et de sous-statut du schéma du **tAssertCatcher**. Pour plus d'informations à propos de cette colonne, consultez le composant *tAssertCatcher, page 1144*.

tAssertCatcher

Propriétés du tAssertCatcher

Famille de composant	Logs & Errors	
Fonction	A partir de son schéma prédéfini, le **tAssertCatcher** retrouve les informations des statuts d'exécution, des exécutions des Jobs et du **tAssert**.	
Objectif	Génère un flux de données consolidant les informations de statut d'une exécution de Job, et transfère les données dans les fichiers de sortie définis.	
Basic settings	*Schema type*	Un schéma est une description de lignes, i.e., il définit les champs à traiter et à passer au composant suivant. Pour ce composant, le schéma est en lecture seule puisqu'il regroupe les informations de log standard, notamment :
		Moment : Date et heure de traitement.
		Pid : ID traités.
		Project : Projet auquel appartiennent les Jobs analysés.
		Job : Nom du Job analysé.
		Language : Langage utilisé par le Job. Cela peut être Java ou Perl.
		Origin : Origine de l'assertion de statut. Elle peut provenir de différents composants **tAssert**.
		Status : Jugements retrouvés à partir du **tAssert**. Ils peuvent être : - **Ok** : si l'instruction conditionnelle du **tAssert** est évaluée comme vraie (*true*) à l'exécution. - **Failed** : si l'instruction conditionnelle du **tAssert** est évaluée comme fausse (*false)*, ou si des erreurs surviennent à l'exécution. Les résultats du Job testé ne correspondent pas à ce qui était attendu, ou une erreur est survenue lors de l'exécution.
		Substatus : Explications détaillées à propos des assertions de statut. Les explications peuvent être : - **Test logically failed** : le Job analysé ne produit pas le résultat attendu. - **Execution Error** : une erreur d'exécution est survenue lors de l'exécution.
		Description : La description textuelle que vous avez saisie dans le **tAssert**.
	Catch Java Exception	Cette case, une fois cochée, vous permet de capturer les erreurs d'exception Java.
	Catch tAssert	Cette case vous permet de capturer les assertions du **tAssert**.

Utilisation	Ce composant est un composant de début de Job secondaire qui retrouve les informations de statut d'exécution à partir de différentes sources. Il génère un flux de données afin de transférer les informations au composant suivant.
Limitation	Ce composant doit être utilisé avec le composant **tAssert**.

Scénario associé

Pour un scénario associé, consultez le scénario du **tAssert** :

- *Scénario : Paramétrer une condition assertive pour l'exécution d'un Job, page 1138*

tChronometerStart

Propriétés du tChronometerStart

Famille de composant	Logs & Errors	
Fonction	Le composant **tChronometerStart** démarre l'opération de mesure du temps d'exécution d'un sous-Job.	
Objectif	Le **tChronometerStart** fonctionne comme un chronomètre : le composant commence à calculer soit le temps de traitement d'un ou plusieurs sous-Jobs du Job principal, soit le temps de traitement d'une partie d'un sous-Job.	
Utilisation	Vous pouvez utiliser le **tChronometerStart** comme composant de début ou au milieu d'un Job. Il peut précéder une ou plusieurs tâches de traitement dans un sous-Job. Il peut également précéder un ou plusieurs sous-Jobs dans le Job principal.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Mesurer la durée de traitement d'un sous-Job ou d'une partie d'un sous-Job*.

tChronometerStop

Propriétés du tChronometerStop

Famille de composant	Logs & Errors	
Fonction	Le composant **tChronometerStop** mesure le temps d'exécution d'un sous-Job.	
Objectif	Le **tChronometerStop** fonctionne comme un chronomètre : arrête le calcul du temps de traitement soit d'un ou de plusieurs sous-Jobs dans le Job principal, soit d'une partie d'un sous-Job. En Perl, le composant **tChronometerStop** affiche le temps d'exécution total, le nombre d'exécutions, le nombre de lignes traitées à la seconde ainsi que le temps minimal, maximal et moyen de traitement d'une ligne. En Java, il affiche le temps d'exécution total.	
Basic settings	*Since options*	Cochez l'une des deux cases pour choisir le point de départ du processus de mesure : **Since the beginning** : arrête le calcul du temps dont la mesure a commencé au début d'un sous-Job. **Since a tChronometerStart** : arrête le calcul du temps dont la mesure a commencé à partir d'un composant **tChronometerStart** utilisé sur le flux de données du sous-Job.
	Display duration in console	Cochez cette case pour afficher les informations d'exécution du sous-Job sur la console.
	Display component name	Cochez cette case pour afficher le nom des composants sur la console.
	Caption	Entrez un texte, pour permettre d'identifier votre sous-Job par exemple.
	Display human readable duration	Cochez cette case pour afficher les informations d'exécution du sous-Job en unités de temps exploitables.
Utilisation	Ne peut être utilisé comme composant de début.	
Limitation	n/a	

Scénario : Mesurer la durée de traitement d'un sous-Job ou d'une partie d'un sous-Job

Ce scénario est un sous-Job, qui, dans une séquence, effectue les opérations suivantes :

- génère 1 000 000 lignes de noms et de prénoms ;
- regroupe les noms et les prénoms correspondants ;
- stocke les données de sortie dans un fichier délimité ;
- mesure la durée d'ensemble du sous-Job ;

- mesure la durée de l'opération de remplacement des noms,
- affiche les informations relatives au temps de traitement dans la console de log **Run**.

Pour mesurer le temps d'exécution du sous-Job :

- A partir de la **Palette**, cliquez-déposez les composants suivants dans l'éditeur graphique : le **tRowGenerator**, le **tMap**, le **tFileOutputDelimited** et le **tChronometerStop**.
- Reliez les trois premiers composants à l'aide de liens de type **Main Row**.

Lorsque vous reliez le **tMap** au **tFileOutputDelimited**, vous serez invités à donner un nom à la table en sortie. Le nom utilisé dans cet exemple est "new_order".

- Reliez le **tFileOutputDelimited** au **tChronometerStop** à l'aide d'un lien de type **OnComponentOk**.
- Sélectionnez le **tRowGenerator** puis cliquez sur la vue **Component** pour paramétrer ses propriétés de base, définies par défaut, dans l'onglet **Basic settings**.

- Cliquez sur le bouton **[...]** du champ **Edit schema** pour définir le schéma du **tRowGenerator**. Cliquez deux fois sur le bouton **[+]** pour ajouter deux colonnes : *First_Name* et *Last_Name*. Cliquez sur **OK** afin de fermer la boîte de dialogue.
- Cliquez sur le bouton **[...]** du champ **RowGenerator Editor** pour ouvrir l'éditeur et définir les données à générer.

- Dans le champ **Number of Rows for RowGenerator** de l'éditeur **[RowGenerator Editor]**, précisez le nombre de lignes à générer puis cliquez sur **OK**. L'éditeur **[RowGenerator Editor]** se ferme.
- Une boîte de dialogue s'ouvre et vous demande si vous souhaitez propager les changements. Cliquez sur **Yes** (Oui).
- Double-cliquez sur le composant **tMap** pour ouvrir son éditeur. L'éditeur du **tMap** affiche le schéma du component **tRowGenerator**.

- Dans l'onglet **Schema editor** de l'éditeur du **tMap**, cliquez sur le bouton **[+]** de la table de sortie pour ajouter deux lignes puis configurez-les.
- Dans l'éditeur du **tMap**, glissez la ligne *First_Name* de la table d'entrée dans la ligne *Last_Name* de la table de sortie et glissez la ligne *Last_Name* dans la ligne *First_Name* de la table de sortie.
- Cliquez sur **Apply** pour enregistrer les modifications.
- A nouveau, une boîte de dialogue s'ouvre et vous demande si vous souhaitez propager les changements. Cliquez sur **Yes**.
- Cliquez sur **OK** pour fermer l'éditeur.

- Sélectionnez le **tFileOutputDelimited** et cliquez sur la vue **Component** pour paramétrer ses propriétés de base dans l'onglet **Basic settings** :

- Sélectionnez le **tChronometerStop** et cliquez sur la vue **Component** pour paramétrer ses propriétés de base dans l'onglet **Basic settings**.
- Dans la zone **Since options,** cochez l'option **Since the beginning** pour mesurer la durée d'exécution du sous-Job entier.

- Cochez/décochez les autres cases en fonction de vos besoins. Dans ce scénario, l'objectif est d'afficher dans la console la durée d'exécution du sous-Job, précédée du nom du composant.
- Si nécessaire, saisissez du texte de le champ **Caption**.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job tChronometerStop at 11:40 05/03/2010.

[statistics] connecting to socket on port 3399
[statistics] connected
[ tChronometerStop_1 ]     2seconds   duration of the subjob
2968 milliseconds
[statistics] disconnected
Job tChronometerStop ended at 11:40 05/03/2010. [exit code=0]
```

Vous pouvez aussi mesurer la durée du sous-Job en plaçant un **tChronometerStop** en dessous du **tRowGenerator**, et en les connectant via un lien **OnSubJobOk**.

tDie

Propriétés du tDie

Les composants **tDie** et **tWarn** sont étroitements liés au composant **tLogCatcher**. Ils sont généralement utilisés ensemble afin que les données de log collectées par le tLogCatcher soient rassemblées et envoyées vers la sortie définie.

Famille de composant	Logs & Errors	
Fonction	Le composant tDie tue le Job en cours d'exécution. Il est généralement utilisé avec un composant tCatch afin de collecter des données de log.	
Objectif	Ce composant déclenche le composant **tLogCatcher** afin de collecter des données de log exhaustives avant de tuer le Job.	
Basic settings	*Die message*	Saisissez le message qui sera affiché avant que le Job ne soit tué.
	Error code	Saisissez le code de l'erreur (un entier) si nécessaire.
	Priority	Sélectionnez le niveau de priorité.
Utilisation	Ce composant ne peut être utilisé en tant que composant de début.	
Limitation	n/a	

Scénarios associés

Pour des exemples d'utilisation du composant **tDie**, consultez les scénarios du composant **tLogCatcher** :

- *Scénario 1 : Avertissement & log lors de la création d'entrées, page 1159*
- *Scénario 2 : Monitorer & arrêter un Job, page 1161*

tFlowMeter

Propriétés du tFlowMeter

Famille de composant	Logs & Errors	
Fonction	Le composant tFlowMeter compte le nombre de lignes traitées dans le flux spécifié.	
Objectif	Le nombre de lignes est ensuite récupéré par le composant **tFlowMeterCatcher** afin d'être ajouté dans les données de log.	
Basic settings	*Use input connection name as label*	Cochez cette case pour réutiliser le nom donné au flux d'entrée Main Row comme libellé des données loggée.
	Mode	Sélectionnez le type de valeur des données mesurées : **Absolute** : le nombre actuel de lignes est loggé. **Relative** : un pourcentage (%) du nombre de lignes est loggé. Lorsque cette option est sélectionnée, la liste **Connections List** vous permet de sélectionner une connexion de référence.
	Thresholds	Ajoutez un seuil pour observer les proportions en volumes mesurés. Vous pouvez décider que le flux doit être compris entre un nombre minimum et maximum de lignes, et si le flux est inférieur au seuil minimum, il y a un goulot d'étranglement.
Utilisation	Ce composant ne peut être utilisé comme composant de début puisqu'il nécessite un flux d'entrée pour fonctionner.	
Limitation	n/a	

Si vous avez besoin d'utiliser les données de log, les statistiques et tout autre mesure des flux de données, consultez *Automatiser l'utilisation de statistiques & logs* du **Guide utilisateur** de **Talend Open Studio**.

Scénario associé

Pour plus d'informations sur l'utilisation du tFlowMeter, consultez le *Scénario : Mesurer le flux d'un Job, page 1155*

tFlowMeterCatcher

Propriétés du tFlowMeterCatcher

Famille de composant	Logs & Errors	
Fonction	Le composant **tFlowMeterCatcher** est basé sur le schéma défini. Il récupère les données du composant **tFlowMeter** et les transmet au composant de sortie.	
Objectif	Ce composant fonctionne comme une fonction de log déclenchée par l'utilisation du composant **tFlowMeter** dans le Job.	
Basic settings	*Schema type*	Un schéma est une description de lignes, i.e., il définit les champs à traiter et à passer au composant suivant. Pour ce composant, le schéma est en lecture seule puisqu'il regroupe les informations de log standard, notamment :
		Moment : Date et heure du traitement
		Pid : Identifiant système du Job.
		Father_pid : Identifiant système du Job parent, le cas échéant. Sinon, le Pid est dupliqué.
		Root_pid : Identifiant système du Job racine, le cas échéant. Sinon, le Pid est dupliqué.
		System_pid : Identifiant système généré par le système.
		Project : Nom du projet auquel appartient le Job.
		Job : Nom du Job en cours.
		Job_repository_id : Identifiant généré par l'application.
		Job_version : Numéro de version du Job en cours.
		Context : Nom du contexte en cours.
		Origin : Nom du composant, le cas échéant.
		Label : Libellé de la connexion main Row précédent le composant **tFlowMeter** dans le Job et dont la volumétrie sera analysée.
		Count : Nombre de lignes en cours de traitement.
		Reference : Nom de la ligne de référence telle qu'elle a été définie dans le composant **tFlowMeter** lorsque le mode Relative a été sélectionné.
		Thresholds : cette fonction ne s'utilise que si le mode Relative a été sélectionné dans le composant **tFlowMeter**.
Utilisation	Ce composant est le composant de début du Job secondaire qui se déclenche automatiquement à la fin du Job principal.	

Limitation	L'utilisation de ce composant est liée à l'utilisation du composant **tFlowMeter**. Pour plus d'informations, consultez *tFlowMeter, page 1153.*

Scénario : Mesurer le flux d'un Job

Le Job suivant permet de recueillir le nombre de lignes traitées. Les mesures sont prises deux fois, une fois après le composant d'entrée, c'est-à-dire avant qu'elles soient filtrées, et après leur filtrage, avant le composant de sortie.

- Cliquez et déposez les composants suivants : **tMysqlInput**, **tFlowMeter** (x2), **tMap**, **tLogRow**, **tFlowMeterCatcher** et **tFileOutputDelimited**.
- Connectez les composants du Job principal à l'aide de liens de type Row main et cliquez sur le libellé pour lui donner un nom plus parlant, par exemple *US_States* après le composant d'entrée et *filtered_states* après le **tMap**.
- Reliez aussi les composants **tFlowMeterCatcher** et **tFileOutputCSV** à l'aide d'une connexion de type Row main puisque les données sont transmises au composant de sortie.
- Dans la vue **Component** du composant **tMysqlInput**, sélectionnez l'option **Repository** dans le champ **Property Type**, si les propriétés de connexion à la base de données sont stockées dans une métadonnée du Repository. Sinon, sélectionnez l'option **Built-in** et paramètrez les propriétés et le schéma manuellement.

- Les 50 Etats des Etats Unis sont enregistrés dans la table *states*. Pour que les 50 entrées soient sélectionnées, la requête à effectuer dans la base de données Mysql est la suivante : `select * from states`.
- Dans l'onglet **Advanced settings**, sélectionnez l'encodage adéquat.
- Puis sélectionnez le composant suivant, le **tFlowMeter**, et paramétrez ses propriétés.

- Cochez la case **Use input connection name as label** pour réutiliser le nom de la connexion d'entrée dans le fichier log de sortie (**tFileOutputCSV**).
- Dans le champ **Mode**, sélectionnez l'option **Absolute** puisqu'il n'y a pas de flux de référence, et aucun seuil ne doit être défini dans le champ **Threshold** pour cet exemple.

Les informations du champ **Thresholds** sont utilisées dans un outil de monitoring tel que l'**Activity Monitoring Console** de **Talend** afin d'obtenir une représentation proportionnelle du flux traité. Pour plus d'informations, consultez le Guide utilisateur de l'**Activity Monitoring Console**.

- Puis lancez l'éditeur du **tMap** pour paramétrer les propriétés de filtrage.
- Pour cet exemple, glissez et déposez les colonnes ID et State de la zone Input (à gauche) du tMap vers la zone Output (à droite). Aucune variable n'est utilisée pour cet exemple.

- Dans la table du flux de sortie (ici, la table *filtered_states*), cliquez sur la flèche dotée d'un (+) pour ajouter un ligne de filtre.
- Glissez la colonne *State* de la zone Input (*row2*) vers le champ filtre et saisissez un filtre permettant de n'obtenir que les Etats commençant par la lettre *M*. L'expression finale ressemble à : `row2.State.startsWith("M")`
- Cliquez sur **OK** pour valider les paramètres.
- Puis sélectionnez le deuxième composant **tFlowMeter** et paramétrez ses propriétés.

- Cochez la case **Use input connection name as label**.
- Dans le champ **Mode**, sélectionnez l'option **Relative** et dans la liste déroulante **Reference connection**, sélectionnez *US_States* comme référence.
- Une fois de plus, aucun seuil n'est utilisé pour cet exemple, ainsi laissez le champ Threshold vide.
- Aucun paramètre particulier n'est requis pour le composant **tLogRow**.
- Il en va de même pour le composant **tFlowMeterCatcher** puisque ses propriétés sont limitées au schéma prédéfini contenant les informations de log.
- Paramétrez les propriétés du composant de sortie (**tFileOutputCSV**).

- Cochez la case **Append** pour logger toutes les mesures du **tFlowMeter**.
- Enregistrez le Job et exécutez-le.

```
Starting job FlowMeterCatcher at 17:56 29/08/2007.
19|Maine
20|Maryland
21|Massachusetts
22|Michigan
23|Minnesota
24|Mississippi
25|Missouri
26|Montana
Job FlowMeterCatcher ended at 17:56 29/08/2007. [exit code=0]
```

La vue **Run** affiche le nom des Etats filtrés tel qu'il a été défini dans le Job.

K	L	M	N	O
origin	label	count	reference	thresholds
1 tFlowMeter_1	US_States	50		
1 tFlowMeter_2	filtered_states	8	50	

Dans le fichier csv délimité, le nombre de lignes de la colonne **count** varie entre le composant **tFlowMeter1** et le **tFlowMeter2** puisque les données ont été filtrées. La colonne **reference** affiche cette différence.

tLogCatcher

Propriétés du tLogCatcher

Les composants **tDie** et **tWarn** sont étroitements liés au composant **tLogCatcher**. Ils sont généralement utilisés ensemble afin que les données de log collectées par le tLogCatcher soient rassemblées et envoyées vers la sortie définie.

Famille de composant	Logs & Errors	
Fonction	Le composant **tLogCatcher** récupère les champs et les messages des fonctions Java Exception/PerlDie et des composants **tDie** et/ou **tWarn** et les transmet au composant suivant.	
Objectif	Ce composant fonctionne comme une fonction de log déclenchée par une des trois fonctions suivantes : Java exception/PerlDie, **tDie** ou **tWarn**, pour collecter et transférer des données de log.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Catch PerlDie *Catch Java Exception*	Cochez cette case pour déclencher la fonction tCatch lorsque les fonctions PerlDie/Java Exception sont appelées dans le Job.
	Catch tDie	Cochez cette case pour déclencher la fonction tCatch lorsque le composant **tDie** est appelé dans le Job.
	Catch tWarn	Cochez cette case pour déclencher la fonction tCatch lorsque le composant **tWarn** est appelé dans le Job.
Utilisation	Ce composant est le composant de début d'un Job secondaire qui se déclenche automatiquement à la fin du Job principal.	
Limitation	n/a	

Scénario 1 : Avertissement & log lors de la création d'entrées

Dans ce scénario constitué de trois composants, un composant **tRowGenerator** crée des données aléatoires (des id à incrémenter). Le flux d'entrée atteint le composant **tWarn** qui déclenche le sous-Job commençant par le **tLogCatcher**. Ce sous-Job récupère le message d'avertissement ainsi que les informations standard et les transmet au **tLogRow** afin d'afficher les données de log.

- Cliquez et déposez les composants **tRowGenerator**, **tWarn**, **tLogCatcher** et **tLogRow** dans l'espace de modélisation.
- Connectez les composants **tRowGenerator** et **tWarn**.
- Connectez les composants **tLogCatcher** et **tLogRow** séparement.
- Dans l'éditeur du **tRowGenerator**, paramétrez la création des données aléatoires à l'aide de fonction de base :

- Dans la vue **Component** du composant **tWarn**, paramétrez le message d'avertissement dans le champ **Warn message**, le code et le niveau de priorité dans le champ **Priority**. Pour cet exemple, le message est "this is a warning'.
- Pour ce scénario, concaténez le message ci-dessus avec une fonction pour collecter la première valeur de la table Input.

- Dans la vue **Component** du composant **tLogCatcher**, cochez la case **tWarn** pour que le message défini dans le **tWarn** soit récupéré par le sous-Job.
- Cliquez sur **Edit Schema** pour vérifier le schéma utilisé pour la sortie de log. Notez que le log est complet.

Appuyez sur **F6** pour exécuter le Job. Notez que le Log produit est exhaustif.

Scénario 2 : Monitorer & arrêter un Job

Ce scénario utilise les composants **tLogCatcher** et **tDie**. Un composant **tRowGenerator** est connecté à un composant **tFileOutputDelimited** à l'aide d'un lien **Row**. Lorsqu'une erreur survient, le composant **tDie** déclenche le sous-Job contenant le **tLogCatcher** qui affiche les données de log dans la console **Run**.

- Cliquez et déposez les composants nécessaires de la **Palette** : **tRowGenerator**, **tFileOutputDelimited**, **tDie**, **tLogCatcher**, **tLogRow**.
- Dans la vue **Component** du composant **tRowGenerator**, paramétrez les propriétés des données d'entrée qui seront traitées.

- Editez le schéma et paramétrez les colonnes suivantes : *id*, *name*, *quantity*, *flag* et *creation*.
- Dans le champ **Number of rows**, saisissez 0. Cela consituera l'erreur sur laquelle sera basée opération Die.
- Dans la colonne **Values**, définissez les fonctions permettant d'alimenter le flux d'entrée.

- Paramétrez les propriétés du **tFileOutputDelimited** afin qu'il contienne les données transmises. La connexion **Row** provenant du **tRowGenerator** alimente automatiquement le schéma du **tFileOutputDelimited**. Dans le champ **Row separator**, saisissez un point-virgule comme séparateur de champ.
- Connectez le composant de sortie au **tDie** à l'aide d'un lien **Trigger** > **If**. Double-cliquez sur cette nouvelle connexion et paramétrez le if :
  `((Integer)globalMap.get("tRowGenerator_1_NB_LINE")) <=0`
- Puis double-cliquez pour sélectionner et paramétrer les propriétés du composant **tDie** dans l'onglet **Basic settings**.

- Dans le champ **Die message**, saisissez le message à transmettre au **tLogCatcher** avant que le Job ne soit arrêté par la fonction Kill.
- Dans le même Job, cliquez et déposez un **tLogCatcher**, sans pour autant le connecter aux composants précédents, et connectez-le à un composant **tLogRow**.
- Dans l'onglet **Basic settings** du composant **tLogCatcher**, paramétrez ses propriétés. Assurez-vous que la case **tDie** est bien cochée pour pouvoir ajouter un message Die aux informations de Log transmises au composant final.

```
Starting job tLogCatcher2 at 10:10 20/04/2010.

No row defined
2010-04-20
10:10:57|RPa7uU|RPa7uU|RPa7uU|TDQ_EE_MPX_JAVA|tLogCatcher2|Default|5|tDie|tDie_
1|No row defined|4
Job tLogCatcher2 ended at 10:10 20/04/2010. [exit code=4]
```

- Appuyez sur **F6** pour exécuter le Job et notez que le log contient un message en noir et un en rouge.
- Les données de log en noir proviennent du **tDie** et sont transmises par le **tLogCatcher**. Et le message Java Exception en rouge apparaît puisque le Job s'est arrêté.

tLogRow

Propriétés du tLogRow

Famille de composant	Logs & Errors	
Fonction	Le composant tLogRow affiche les données ou les résultats dans la console **Run**.	
Objectif	Ce composant permet de monitorer les données traitées.	
Basic settings	*Print values in table cells*	Le flux de sortie apparaît dans un tableau.
	Separator	Saisissez le séparateur qui délimitera les données dans la console.
	Print component unique name in front of each output row	Cochez cette case si plusieurs composants LogRow sont utilisés. Cela permet de différencier les sorties.
	Print schema column name in front of each value	Cochez cette case pour récupérer les libellés des colonnes du schéma de sortie.
	Use fixed length for values	Cochez cette case pour paramétrer une largeur fixe aux valeurs affichées.
Utilisation	Ce composant peut être utilisé en composant intermédiaire dans un flux de données ou en composant de fin dans un Job design.	
Limitation	n/a	

Scénarios associés

Pour plus d'informations sur l'utilisation du **tLogRow**, consultez les composants :

- **tFileInputDelimited,** *Scénario 1 : Afficher le contenu d'un fichier délimité, page 917*.
- **tContextLoad**, *Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL, page 1190*.
- **tWarn**, **tDie** et **tLogCatcher**, *Scénario 1 : Avertissement & log lors de la création d'entrées, page 1159* et *Scénario 2 : Monitorer & arrêter un Job, page 1161*.

tStatCatcher

Propriétés du tStatCatcher

Famille de composant	Logs & Errors	
Fonction	Le composant tStatCatcher est basé sur le schéma défini et regroupe les métadonnées du Job en cours d'exécution, ainsi que les métadonnées de chaque composant de ce Job.	
Objectif	Ce composant fonctionne comme une fonction de log déclenchée pour chaque composant dont la case StatsCatcher Statistics est cochée. Il collecte et transfert ces données de log à la sortie définie.	
Basic settings	*Schema type*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Pour ce composant, le schéma est en lecture seule, puisqu'il regroupe les informations de log standard, notamment :
		Moment : Date et heure du traitement
		Pid : Identifiant système du Job.
		Father_pid : Identifiant système du Job parent, le cas échéant. Sinon, le Pid est dupliqué.
		Root_pid : Identifiant système du Job racine, le cas échéant. Sinon, le Pid est dupliqué.
		Project : Nom du projet auquel appartient le Job.
		Job : Nom du Job en cours.
		Context : Nom du contexte en cours.
		Origin : Nom du composant, le cas échéant.
		Message : Informations de début et de fin du Job.
Utilisation	Ce composant est le composant de début d'un Job secondaire qui se déclenche automatiquement à la fin du Job principal. Les dates de début et de fin apparaissent dans le log.	
Limitation	n/a	

Scénario : Afficher les statistiques du Job

Ce scénario décrit un Job à quatre composants, permettant d'afficher dans la console **Run** les statistiques recueillies lors de la génération du fichier grâce au composant tStatCatcher.

- Cliquez et déposez les composants : **tRowGenerator**, **tFileOutputDelimited**, **tStatCatcher** et **tLogRow**
- Dans l'onglet **Basic settings** du composant **tRowGenerator**, paramétrez les données à générer. Pour ce Job, le schéma est composé de trois colonnes : *ID_Owners*, *Name_Customer* et *ID_Insurance*, générées à l'aide d'un script Perl.

- Le nombre de lignes peut être limité à 100.
- Cliquez sur l'onglet **Main** de la vue **Component**.

- Et cochez la case **tStatCatcher Statistics** pour activer la fonction de récupération des statistiques.
- Puis, paramétrez les propriétés du composant de sortie. Dans la vue **Component** du **tFileOutputDelimited**, sélectionnez le fichier de sortie ou saisissez le nom du fichier de sortie à créer. Paramétrez les séparateurs de champs (point-virgule) et l'encodage.
- Cliquez sur **Edit schema** et assurez-vous que le schéma est récupéré du schéma d'entrée. Si nécessaire, cliquez sur **Sync Columns** pour récupérer le schéma du composant précédent.

- Puis cliquez sur l'onglet **Advanced settings** de la vue **Component**, et cochez la case **tStatCatcher Statistics** pour activer la fonction de récupération des statistiques.
- Dans le deuxième Job, double-cliquez sur le composant **tStatCatcher**. Notez que les propriétés sont fournis uniquement pour information puisque le schéma représentant les données à regrouper et agréger en statistiques, est en lecture seule.

Column	Key	Type	Nullable	Length	Precision	Comment
moment		Day	✓			
pid		String	✓	20		
father_pid		String	✓	20		
root_pid		String	✓	20		
project		String	✓	50		
job		String	✓	50		
context		String	✓	50		
origin		String	✓	255		
message		String	✓	255		

- Paramétrez ensuite les propriétés du **tLogRow**. Définissez le séparateur de données qui apparaîtra dans la console.
- Enfin, appuyez sur **F6** pour exécuter le Job et afficher les résultats du Job.

```
Starting job StatsCatch at 15:10 23/02/2007.
2007-02-23 15:10:30|3656|StatsCatch|Default||begin
2007-02-23
15:10:30|3656|StatsCatch|Default|tFileOutputDelimited_1|begin
2007-02-23 15:10:30|3656|StatsCatch|Default|tRowGenerator_1|begin
2007-02-23 15:10:30|3656|StatsCatch|Default|tRowGenerator_1|end
2007-02-23 15:10:30|3656|StatsCatch|Default|tRowGenerator_1|0.0 seconds
2007-02-23 15:10:30|3656|StatsCatch|Default|tFileOutputDelimited_1|end
2007-02-23 15:10:30|3656|StatsCatch|Default|tFileOutputDelimited_1|0.0
seconds
2007-02-23 15:10:30|3656|StatsCatch|Default||end
2007-02-23 15:10:30|3656|StatsCatch|Default||0.0 seconds
Job StatsCatch ended at 15:10 23/02/2007. [exit code=0]
```

Les données de log affiche les informations de début et de fin du Job en lui-même, ainsi que pour chaque composant.

tWarn

Propriétés du tWarn

Les composants **tDie** et **tWarn** sont étroitements liés au composant **tLogCatcher**. Ils sont généralement utilisés ensemble afin que les données de log collectées par le tLogCatcher soient rassemblées et envoyées vers la sortie définie.

Famille de composant	Logs & Errors	
Fonction	Le composant **tWarn** envoie un message d'avertissement au composant suivant.	
Objectif	Ce composant déclenche un avertissement souvent pris comme données de log exhaustives par le composant **tLogCatcher**.	
Basic settings	*Warn message*	Saisissez votre message d'avertissement.
	Code	Saisissez le niveau de code.
	Priority	Sélectionnez le niveau de priorité.
Utilisation	Ce composant ne peut être utilisé comme composant de début. Ainsi, s'il est connecté à un composant de sortie, il doit aussi être connecté à un composant d'entrée.	
Limitation	n/a	

Scénarios associés

Pour des exemples d'utilisation du **tWarn**, consultez les scénarios du composant **tLogCatcher** :

- *Scénario 1 : Avertissement & log lors de la création d'entrées, page 1159*
- *Scénario 2 : Monitorer & arrêter un Job, page 1161*

Composants Misc

Ce chapitre passe en revue les principaux composants présents dans la famille **Misc** de la Palette de **Talend Open Studio**.

La famille **Misc** regroupe divers connecteurs permettant de répondre à des besoins, tels que la création de lignes de données factices, la mise en tampon de données, le chargement de variables de contexte.

tAddLocationFromIP

Propriétés du tAddLocationFromIP

Famille du composant	Misc	
Fonction	Le composant **tAddLocationFromIP** remplace des adresses IP par le lieu géographique correspondant.	
Objectif	Le **tAddLocationFromIP vous permet de localiser géographiquement les visiteurs grâce à leurs adresses IP : il identifie le lieu géographique des visiteurs (pays, région, ville,** latitude, longitude, code postal, etc.) en utilisant un fichier de consultation d'adresse IP.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé pour ce composant uniquement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisable dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Database Filepath	Chemin d'accès au fichier de consultation d'adresses IP.
	Input parameters	**Input column :** Sélectionnez la colonne d'entrée à partir de laquelle les valeurs sont collectées pour l'agrégation.
		input value is a hostname : Cochez cette case si la colonne d'entrée comporte des noms d'hôtes.
		input value is an IP address : Cochez cette case si la colonne d'entrée comporte des adresses IP.
	Location type	**Country code** : Cochez cette case pour remplacer l'adresse IP par le code du pays correspondant.
		Country name : Cochez cette case pour remplacer l'adresse IP par le nom du pays correspondant.
Utilisation	Ce composant peut être utilisé comme étape intermédiaire du traitement permettant de remplacer des adresses IP par des informations géolocalisées. Il ne peut être utilisé comme composant de début car il nécessite un flux en entrée, ainsi qu'un composant de sortie.	
Limitation	n/a	

Scénario : Identifier la localisation géographique d'une adresse IP

Le scénario Java suivant crée un Job à trois composants associant une adresse IP à un lieu géographique. Il permet d'obtenir la localisation géographique du visiteur d'un site à partir de son adresse IP.

- A partir de la Palette, cliquez-déposez les composants suivants dans l'éditeur graphique : le **tFixedFlowInput**, le **tAddLocationFromIP**, et le **tLogRow**.
- Connectez les trois composants en utilisant des liens de type **Row Main**.

- Dans l'espace graphique, sélectionnez le **tFixedFlowInput**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tFixedFlowInput**.
- Configurez le champ **Schema Type** en mode **Built-In** puis cliquez sur le bouton **[...]** à côté du champ **Edit Schema** pour définir les données que vous voulez utiliser en entrée. Dans ce scénario, le schéma est constitué d'une colonne comportant une adresse IP.

- Cliquez sur le bouton **OK** pour fermer la boîte de dialogue, puis acceptez la propagation des modifications lorsque le système vous invite à le faire. La colonne définie s'affiche dans la section **Values** de la vue **Basic settings**.
- Cliquez dans la cellule **Value** puis définissez la valeur de l'adresse IP.

- Dans le champ **Number of rows**, entrez le nombre de lignes à générer.
- Dans l'espace graphique, sélectionnez le composant **tAddLocationFromIP**.
- Cliquez dans la vue **Component** pour définir la configuration de base (**Basic settings**) du **tAddLocationFromIP**.

- Cliquez sur le bouton **Sync columns** pour synchroniser le schéma avec le schéma d'entrée du **tFixedFlowInput**.
- Parcourez vos dossiers jusqu'au fichier GeoIP.dat pour en définir le chemin d'accès dans le champ **Database filepath**.

Assurez-vous de télécharger la dernière version du fichier de consultation des adresses IP, à partir du site indiqué dans la vue **Basic settings** du composant **tAddLocationFromIp**.

- Dans la zone **Input parameters**, définissez les paramètres d'entrée selon vos besoins. Dans ce scénario, la colonne d'entrée correspond à la colonne *ip*, qui comporte une adresse IP, définie précédemment.
- Dans la zone **Location type**, définissez le type de lieu selon vos besoins. Dans ce scénario, on veut afficher le nom du pays.
- Dans l'éditeur graphique, sélectionnez le composant **tLogRow**.
- Cliquez sur la vue **Component** puis définissez la configuration de base (**Basic settings**) du **tLogRow** selon vos besoins. Dans ce scénario, on veut afficher les valeurs dans les cellules d'une table.
- Enregistrez votre Job puis cliquez sur **F6** pour l'exécuter.

```
Starting job add_location at 11:00 06/08/2008.
.------------+--------.
|      tLogRow_1       |
|=------------+-------=|
|ip           |location|
|=------------+-------=|
|88.131.8.250|Sweden  |
'------------+--------'

Job add_location ended at 11:00 06/08/2008. [exit code=0]
```

La ligne unique générée par le Job affiche le nom du pays correspondant à l'adresse IP renseignée.

tBufferInput

Propriétés du tBufferInput

Famille de composant	Misc	
Fonction	Ce composant permet de récupérer les données mises en tampon pour pouvoir les réutiliser dans un deuxième sous-job.	
Objectif	Lee composant tBufferInput récupère les données mises en mémoire tampon, via un composant tBufferOutput par exemple, pour les traiter dans un deuxième temps.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Lors de l'utilisation du **tBufferInput**, c'est l'ordre des colonnes qui est pris en compte, ainsi l'ordre des colonnes est plus important que leur libellé.
		Built-in : Le schéma est créé et conservé pour ce composant uniquement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisable dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
Utilisation	Ce composant est le composant de début d'un Job secondaire qui se déclenche automatiquement à la fin du Job principal.	

Scénario : Récupérer les données mises en tampon (Java)

Ce scénario décrit un Job qui récupère les données mises en mémoire tampon lors de l'exécution d'un premier sous-job et les affiche dans la console standard.

- Cliquez et déposez les composants suivants : **tFileInputDelimited** et **tBufferOutput**.
- Sélectionnez le composant **tFileInputDelimited** et dans l'onglet **Basic Settings** de la vue **Component**, paramétrez les propriétés d'accés au fichier d'entrée.

- Dans le champ **File Name**, paramétrez le chemin d'accès au fichier délimité contenant les données à mettre en tampon.
- Dans les champs **Row** et **Field separators**, renseignez respectivement les séparateurs de lignes et de champs.
- Dans le champ **Header**, renseignez les informations sur un éventuel en-tête.
- Cliquez sur le bouton [...] à coté du champ **Schema type** pour décrire la structure du fichier

- Décrivez le **Schéma** des données à passer au composant **tBufferOutput**.
- Sélectionnez le composant **tBufferOutput** et paramétrez ses propriétés dans l'onglet **Basic Settings** de la vue **Component**.

- Généralement le schéma du composant d'entrée alimente automatiquement le schéma du composant **tBufferOutput**. Mais vous pouvez aussi configurer une partie du schéma à mettre en tampon si vous le souhaitez.
- Déposez les composants **tBufferInput** et **tLogRow** en dessous du sous-job que vous venez de créer.
- Reliez le composant **tFileInputDelimited** au **tBufferInput** via un lien de type **Trigger** > **OnSubjobOk** et reliez le composant **tBufferInput** au **tLogRow** via un lien de type **Row** > **Main**.
- Double-cliquez sur le composant **tBufferInput** pour paramétrer ses propriétés dans la vue **Component**.
- Dans l'onglet **Basic settings**, cliquez sur le bouton [...] à coté du champ **Schema** pour décrire la structure du fichier.

- Utilisez la même description de fichier que celle définie pour le composant **tFileInputDelimited** et cliquez sur **OK**.
- Le schéma du composant **tBufferInput** alimente automatiquement le schéma du composant **tLogRow**. Sinon double-cliquez sur le **tLogRow** pour afficher ses propriétés dans la vue **Component** et cliquez sur le bouton **Sync columns**.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job tBufferInput_scenario at 14:13 12/06/2009.

AL|Alabama|Montgomery|Birmingham
AK|Alaska|Juneau|Anchorage
AZ|Arizona|Phoenix|Phoenix
AR|Arkansas|Little Rock|Little Rock
CA|California|Sacramento|Los Angeles
CO|Colorado|Denver|Denver
CT|Connecticut|Hartford|Bridgeport
DE|Delaware|Dover|Wilmington
FL|Florida|Tallahassee|Jacksonville
GA|Georgia|Atlanta|Atlanta
HI|Hawaii|Honolulu|Honolulu
ID|Idaho|Boise|Boise
IL|Illinois|Springfield|Chicago
IN|Indiana|Indianapolis|Indianapolis
IA|Iowa|Des Moines|Des Moines
KS|Kansas|Topeka|Wichita
KY|Kentucky|Frankfort|Louisville
```

La console affiche les données qui ont été récupérés de la mémoire tampon.

tBufferOutput

Propriétés du tBufferOutput

Famille de composant	Misc	
Fonction	Ce composant met en tampon des données afin de pouvoir y accéder plus tard via un service Web par exemple.	
Objectif	Ce composant permet d'accéder aux données via un service Web. En effet, il a été conçu pour être exporté en tant que service Web afin d'accéder aux données directement à partir du serveur d'application Web. Pour plus d'informations, consultez *Exporter un Job en tant que WebService* du **Guide utilisateur** de **Talend Open Studio**.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Lors de l'utilisation du **tBufferOutput**, c'est l'ordre des colonnes qui est pris en compte, ainsi l'ordre des colonnes est plus important que leur libellé.
		Built-in : Le schéma est créé et conservé pour ce composant uniquement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisable dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
Utilisation	Ce composant n'est pas un composant de début (font vert) et il requiert un composant de sortie.	

Scénario 1 : Mettre des données en tampon (Java)

Ce scénario décrit un Job volontairement basique qui met en tampon les données d'un Job enfant pendant qu'un Job parent affiche les données mise en tampon dans la console standard. Généralement, un Job utilisant un composant **tBufferOutput** incluerait une étape d'export en tant que service Web, afin que les données soient accessibles directement via le serveur d'application Web.

- Créez deux Jobs : le premier Job (*BufferFatherJob*) exécute le deuxième et affiche le contenu dans la console **Run**. Le deuxième Job (*BufferChildJob*) stocke les données définies dans une mémoire tampon.
- Dans le premier Job, cliquez et déposez les composants suivants : **tRunJob** et **tLogRow**.
- Dans le deuxième Job, cliquez et déposez les composants suivants : **tFileInputDelimited** et **tBufferOutput**.
- Commencez par paramétrer les propriétés du deuxième Job.
- Sélectionnez le composant **tFileInputDelimited** et dans l'onglet **Basic Settings** de la vue **Component**, paramétrez les propriétés d'accés au fichier d'entrée.

- Dans le champ **File Name**, paramétrez le chemin d'accès au fichier délimité contenant les données à mettre en tampon.
- Dans les champs **Row** et **Field separators**, renseignez respectivement les séparateurs de lignes et de champs.
- Dans le champ **Header**, renseignez les informations sur un éventuel en-tête.

tFileInputDelimited_2

Column	Key	Type	Nullable	Date Patt...	Length	Pre...	D...	Co...
Postal	☐	String	☑		2			
State	☐	String	☑		14			
Capital	☐	String	☑		14			
MostPopulousCity	☐	String	☑		14			

- Décrivez le **Schéma** des données à passer au composant **tBufferOutput**.
- Sélectionnez le composant **tBufferOutput** et paramétrez ses propriétés dans l'onglet **Basic Settings** de la vue **Component**.

Schema Type Built-In Edit schema Sync columns

- Généralement le schéma du composant d'entrée alimente automatiquement le schéma du composant tBufferOutput. Mais vous pouvez aussi configurer une partie du schéma à mettre en tampon si vous le souhaitez.
- Maintenant le premier Job (*BufferFatherJob*), paramétrez les propriétés du composant **tRunJob**.

Schema Type Built-In Edit schema
☑ Die on child error
Job /v23a/Buffering/BufferChildJob * Context Defa
Context Param

Parameters	Values

- Cliquez sur **Edit schema** pour modifier le schéma si nécessaire et sélectionner la colonne à afficher. Le schéma peut être identique ou différent du schéma mis en tampon.
- Dans le tableau **Context Param**, vous pouvez aussi définir les paramètres de contexte à utiliser pour l'exécution de ce Job. Pour cet exemple, gardez le contexte par défaut.

Appuyez sur **F6** pour exécuter les Jobs parent. Le composant **tRunJob** se charge d'exécuter le Job enfant et affiche les données dans la console standard :

```
Starting job BufferFatherJob at 10:41 05/02/2008.
AL|Alabama|Montgomery|Birmingham
AK|Alaska|Juneau|Anchorage
AZ|Arizona|Phoenix|Phoenix
AR|Arkansas|Little Rock|Little Rock
CA|California|Sacramento|Los Angeles
CO|Colorado|Denver|Denver
CT|Connecticut|Hartford|Bridgeport
DE|Delaware|Dover|Wilmington
FL|Florida|Tallahassee|Jacksonville
GA|Georgia|Atlanta|Atlanta
HI|Hawaii|Honolulu|Honolulu
ID|Idaho|Boise|Boise
IL|Illinois|Springfield|Chicago
IN|Indiana|Indianapolis|Indianapolis
IA|Iowa|Des Moines|Des Moines
KS|Kansas|Topeka|Wichita
KY|Kentucky|Frankfort|Louisville
```

Scénario 2 : Mettre les données de sortie en mémoire tampon du serveur d'application Web

Ce scénario décrit un Job appelant un service Web et stockant les données de sortie directement sur la mémoire tampon du serveur d'application Web. Ce scénario crée d'abord un Job utilisant des variables de contexte qui pourra ensuite être utilisé en tant que service Web, puis exporte de Job en tant que service Web.

Créer un Job utilisant des variables de contexte et pouvant être exporté en tant que service Web :

- Cliquez-déposez les composants suivants dans le Job Designer : **tFixedFlowInput** et **tBufferOutput**.
- Reliez-les via une connexion de type **Row Main**.

- Dans l'espace de travail, sélectionnez le **tFixedFlowInput**.
- Cliquez sur la vue **Component** pour paramétrer ses propriétés de base dans l'onglet **Basic settings**.
- Dans la liste **Schema Type**, sélectionnez **Built-In** et cliquez sur le bouton [...] à coté du champ **Edit Schema** pour décrire la structure des données que vous souhaitez à partir des variables internes. Dans ce scénario, le schéma est constitué de trois colonnes : *now*, *firstname*, et *lastname*.

- Cliquez trois fois sur le bouton (+) pour ajouter trois lignes de paramètre et configurez vos variables.
- Cliquez sur **OK** pour fermer la boîte de dialogue et acceptez la propagation des modifications lorsque vous y êtes invité. Les trois colonnes définies apparaissent dans le champ **Values** de l'onglet **Basic settings** du **tFixedFlowInput**.

- Cliquez dans le cellule **Value** de chaque ligne des deux premier paramètres définis et appuyez sur **Ctrl+Space** pour accéder à la liste des variables globales.
- A partir de la liste des variables globales, sélectionnez *TalendDate.getCurrentDate()* pour la colonne *now* et *talendDatagenerator.getFirstName* pour la colonne *firstname*.

Pour ce scénario, l'objectif est de définir deux variables de contexte : *nb_lines* et *lastname*. Dans la première, paramétrez le nombre de lignes à générer, et dans le deuxième, paramétrez les noms qui apparaitront dans la liste de sortie. Le composant **tFixedFlowInput** générera le nombre de lignes et les trois colonnes : *now*, *firstname*, et *lastname* définies dans les variables de contexte. Pour plus d'informations concernant l'utilisation des variables de contexte : consultez *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.

Pour définir les deux variables de contexte :

- Sélectionnez le composant **tFixedFlowInput** et cliquez sur la vue **Contexts**.
- Dans la vue **Variables**, cliquez sur le bouton **[+]** pour ajouter deux lignes de paramètres et configurez-les.

- Cliquez sur l'onglet **Values as table** et configurez le premier paramètre définissant le nombre de lignes à générer et le second paramètre définissant les noms à afficher.

- Cliquez sur la vue **Component** pour afficher l'onglet **Basic settings** du composant **tFixedFlowInput**.
- Cliquez dans la cellule **Value** de la colonne *lastname* et appuyez sur **Ctrl+Space** pour accéder à la liste des variables globales.
- A partir de la liste des variables globales, sélectionnez *context.lastname*, la variable de contexte que vous avez créé pour la colonne *lastname*.

Exporter un Job en tant que service Web :

Avant d'exporter votre Job en tant que service Web, consultez la section *Export de Job en Java* du Guide utilisateur de **Talend Open Studio**.

- Dans la vue Repository, cliquez-droit sur le Job que vous venez de créer et sélectionnez **Export Job Scripts** dans le menu. La boîte de dialogue **[Export Job Scripts]** s'ouvre.

- Cliquez sur le bouton **Browse...** et sélectionnez le répertoire dans lequel enregistrer votre Job.
- Dans la zone **Export type**, sélectionnez le type d'export que vous souhaitez utiliser dans votre application Web Tomcat (WAR pour cet exemple) et cliquez sur **Finish**. La boîte de dialogue **[Export Job Scripts]** se referme.
- Copiez le fichier War et collez-le dans le répertoire de votre Tomcat.

Scénario 3 : Appeler un Job contenant des variables de contexte à partir de votre navigateur Web

Ce scénario décrit la procédure à utiliser pour appeler le Job créé dans le scénario 2 à partir de votre navigateur Web en modifiant/sans modifier les valeurs des variables de contexte.

Saisissez l'URL suivante dans votre navigateur Web : *http://localhost:8080//export_job/services/export_job3?method=runJob* dans laquelle "export_job" correspond au nom du répertoire de votre application Web déployé sur le Tomcat et "export_job3" correspond au nom de votre Job.

Cliquez sur **Enter** pour exécuter votre Job à partir du navigateur.


```
- <soapenv:Envelope>
  - <soapenv:Body>
    - <runJobReturn xsi:type="ns1:runJobReturn">
      - <ns1:item xsi:type="ns1:ArrayOf_xsd_string">
          <ns1:item xsi:type="xsd:string">31-07-2008</ns1:item>
          <ns1:item xsi:type="xsd:string">William</ns1:item>
          <ns1:item xsi:type="xsd:string">Ford</ns1:item>
        </ns1:item>
      - <ns1:item xsi:type="ns1:ArrayOf_xsd_string">
          <ns1:item xsi:type="xsd:string">31-07-2008</ns1:item>
          <ns1:item xsi:type="xsd:string">Millard</ns1:item>
          <ns1:item xsi:type="xsd:string">Ford</ns1:item>
        </ns1:item>
      - <ns1:item xsi:type="ns1:ArrayOf_xsd_string">
          <ns1:item xsi:type="xsd:string">31-07-2008</ns1:item>
          <ns1:item xsi:type="xsd:string">James</ns1:item>
          <ns1:item xsi:type="xsd:string">Ford</ns1:item>
        </ns1:item>
      </runJobReturn>
    </soapenv:Body>
  </soapenv:Envelope>
```


Le Job utilise les valeurs par défaut des variables de contexte : *nb_lines* et *lastname*, et donc génère trois lignes contenant la date en cours, un prénom et le nom Ford.

Vous pouvez modifier les valeurs des variables de contexte directement à partir de votre navigateur. Pour appeler le Job à partir de votre navigateur et modifier les valeurs des deux variables de contexte, saisissez l'URL suivante :

http://localhost:8080//export_job/services/export_job3?method=runJob&arg1=--context_param%20lastname=MASSY&arg2=--context_param%20nb_lines=2.

%20 correspond à un espace blanc lorsqu'il est utilisé dans une URL. Dans le premier argument "arg1", définissez la valeur "MASSY" pour la variable de contexte *lastname*. Ainsi le nom qui apparaîtra sera "MASSY". Dans le deuxième argument "arg2", définissez la valeur "2" pour la variable de contexte *nb_lines* pour ne générer que deux lignes de données.

Cliquez sur **Enter** pour exécuter le Job à partir du navigateur.

```
- <soapenv:Envelope>
  - <soapenv:Body>
    - <runJobReturn xsi:type="ns1:runJobReturn">
      - <ns1:item xsi:type="ns1:ArrayOf_xsd_string">
          <ns1:item xsi:type="xsd:string">31-07-2008</ns1:item>
          <ns1:item xsi:type="xsd:string">Richard</ns1:item>
          <ns1:item xsi:type="xsd:string">MASSY</ns1:item>
        </ns1:item>
      - <ns1:item xsi:type="ns1:ArrayOf_xsd_string">
          <ns1:item xsi:type="xsd:string">31-07-2008</ns1:item>
          <ns1:item xsi:type="xsd:string">Theodore</ns1:item>
          <ns1:item xsi:type="xsd:string">MASSY</ns1:item>
        </ns1:item>
      </runJobReturn>
    </soapenv:Body>
  </soapenv:Envelope>
```

Le Job génère deux lignes avec MASSY comme nom.

Scénario 4 : Appeler un Job exporté en tant que service Web dans un autre Job

Ce scénario décrit un Job appelant un autre Job exporté en tant que service Web en utilisant un composant **tWebServiceInput**. Ce scénario appelera le Job créé dans le scénario 2.

- Cliquez-déposez les composants suivants dans le Job Designer : **tWebServiceInput** et **tLogRow**.
- Reliez-les via une connexion de type **Row Main**.

- Dans l'espace de travail, sélectionnez le composant **tWebServiceInput**.
- Cliquez sur la vue **Component** pour définir ses propriétés de base dans l'onglet **Basic settings**.

- Dans la liste **Schema Type**, sélectionnez **Built-In** et cliquez sur le bouton [...] à coté du champ **Edit Schema** pour décrire la structure des données que vous souhaitez appeler du Job exporté. Dans ce scénario, le schéma est constitué de trois colonnes, *now*, *firstname*, et *lastname*.

- Cliquez sur le bouton (+) pour ajouter les trois lignes de paramètres et définir vos variables. Cliquez sur **OK** pour fermer la boîte de dialogue.
- Dans le champ **WSDL** de l'onglet **Basic settings** de la vue **Component** du **tWebServiceInput**, saisissez l'URL : *http://localhost:8080/export_job/services/export_job3?WSDL* dans laquelle "export_job" correspond au nom du répertoire de votre application Web dans lequel le Job à appeler est stocké et "export_job3" correspond au nom du Job.

- Dans le champ **Method name**, saisissez *runJob*.
- Dans la zone **Parameters**, cliquez sur le bouton (+) pour ajouter deux lignes de paramètres et définir vos variables de contexte.
- Cliquez dans la première cellule **Value** pour configurer le paramètre définissant le nombre de lignes à générer : *--context_param nb_line=3*.
- Cliquez dans la deuxième cellule **Value** pour configurer le maramètre définissant le nom à afficher : *--context_param lastname=Ford*.
- Sélectionnez le **tLogRow** et cliquez sur la vue **Component** pour afficher les propriétés du composant.
- Dans la vue **Basic settings** du **tLogRow**, sélectionnez l'option Table pour afficher les données de sortie sous forme de tableau. Pour plus d'informations, consultez les *Propriétés du tLogRow, page 1163*.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job Call_Webservice_In_Job at 14:12 01/08/2008.
.----------+---------+--------.
|          tLogRow_1           |
|=---------+---------+-------=|
|now       |firstname|lastname|
|=---------+---------+-------=|
|01-08-2008|Ronald   |Ford    |
|01-08-2008|Woodrow  |Ford    |
|01-08-2008|Franklin |Ford    |
'----------+---------+--------'

Job Call_Webservice_In_Job ended at 14:12 01/08/2008. [exit code=0]
```

Le Job génère trois colonnes contenant la date en cours, des prénoms et des noms, et les affiche dans la console sous forme de tableau.

tContextDump

Propriétés du tContextDump

Famille de composant	Misc	
Fonction	**tContextDump** fait une copie des valeurs du contexte utilisé dans le Job.	
Objectif	Le **tContextDump** peut être utilisé pour transformer les paramètres de contexte d'un flux et les réutilisez via un composant **tContextLoad**. Cette fonction est très pratique pour réutiliser un contexte que vous n'aurez défini qu'une seule fois, dans plusieurs Jobs via le **tContextLoad**..	
Basic settings	*Schema type* et *Edit Schema*	Le schéma du composant **tContextDump** est en lecture seule et est composé de deux colonnes : Key et Value, qui correspondent au nom et à la valeur du paramètre à copier. Un schéma est une description de lignes, i.e., il définit les champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur Edit Schema pour modifier le schéma. Notez que lorsque vous modifiez le schéma, les propriétés deviennent automatiquement built-in.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Print operations	Cochez cette case pour afficher les paramètres de contexte dans la vue **Run**.
Utilisation	Ce composant crée un flux de données à partir des valeurs de contexte, ainsi il doit être connecté à un composant de sortie.	
Limitation	Le **tContextDump** ne crée pas de variable de contexte non définie.	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tContextLoad

Propriétés du tContextLoad

Famille de composant	Misc	
Fonction	Le composant **tContextLoad** modifie dynamiquement les valeurs du contexte actif.	
Objectif	Le **tContextLoad** permet de charger un contexte à partir d'un flux entrant. Ce composant effectue aussi deux contrôles. Il prévient lorsque les paramètres définis dans le flux entrant ne sont pas définis dans le contexte. Il avertit également lorsque la valeur d'un contexte n'est pas initialisée dans le flux entrant. Mais notez que ces avertissements ne bloquent pas le traitement.	
Basic settings	*Schema type* et *Edit Schema*	Le schéma du composant **tContextLoad** doit être composé de deux colonnes, l'une contenant le nom du paramètre et la deuxième contenant la valeur du paramètre à charger. Un schéma est une description de lignes, i.e., il définit le nombre de champ qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, les propriétés deviennent automatiquement built-in.
		Built-in : Le shéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** Talend Open Studio.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de Talend Open Studio.
	If a variable loaded, but not in the context	Si une variable est chargée mais n'apparaît pas dans le contexte, choisissez comment doit s'afficher la notification. Sous forme d'erreur (**Error**), de warning (**warning**), ou d'information (**info**).
	If a variable in the context, but not loaded	Si une variable apparaît dans le contexte mais n'est pas chargée, choisissez comment doit s'afficher la notification. Sous forme d'erreur (**Error**), de warning (**warning**), ou d'information (**info**).
	Print operations	Cochez cette case pour afficher les paramètres de contexte dans la vue **Run**.
	Disable errors	Cochez cette case pour ne pas afficher d'erreur.

	Disable warnings	Cochez cette case pour ne pas afficher de warning.
	Disable infos	Cochez cette case pour ne pas afficher d'information.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et les lignes contenant les erreurs seront ignorées.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant s'appuie sur le flux de données pour charger les valeurs de contexte à utiliser. Ainsi, il requiert un composant d'entrée et ne peut pas être un composant de début.	
Limitation	Le **tContextLoad** ne crée aucune variable qui n'a pas été définie dans le contexte par défaut.	

Scénario : Utiliser un contexte dynamique lors d'un insert dans une base de données MySQL

Ce scénario est composé de deux sous-Jobs. Le premier permet de charger dynamiquement des paramètres de contexte et le deuxième utilise ces paramètres chargés pour afficher le contenu d'une base de données.

- Cliquez et déposez les composants : **tFilelist**, **tFileInputDelimited**, **tContextLoad** pour le premier sous-Job.
- Et cliquez et déposez les composants **tMysqlInput** et **tLogRow** pour le deuxième.
- Connectez-les comme indiqué sur le schéma ci-dessus.
- Créez autant de fichiers délimités qu'il y a de contextes différents et conservez-les dans un répertoire spécifique appelé *Contexts*. Dans ce scénario, *test.txt* contient les informations de connexion à la base de données locale de test et *prod.txt* contient les informations de connexion de la base de données distante de production.
- Chaque fichier est constitué de deux champs contenant le nom du paramètre et sa valeur correspondante, en fonction du contexte.

- Dans l'onglet **Basic settings** du composant **tFileList**, sélectionnez le répertoire *contexts* dans lequel les deux fichiers *test* et *prod* seront stockés.
- Dans l'onglet **Basic settings** du composant **tFileInputDelimited**, dans le champ **File Name**, appuyez sur **Ctrl+Espace** pour accéder à la liste des variables globales. Sélectionnez `tFileList_1.CURRENT_FILEPATH` qui permet de faire une boucle sur le répertoire contenant les fichiers de contexte.
- Définissez le schéma manuellement (Built-in) en cliquant sur le bouton **Edit schema**. Il contient deux colonnes : *Key* et *Value*, correspondant respectivement au paramètre et sa valeur.
- Acceptez la propagation du schéma défini au composant suivant (**tContextLoad**).
- Pour ce scénario, cochez la case **Print operations** pour afficher les paramètres de contexte utilisés dans la vue **Run**.
- Puis double-cliquez sur le composant **tMysqlInput** pour accéder à ses propriétés dans l'onget **Basic settings**.
- Pour chacune des valeurs stockées dans le fichier de contexte, appuyez sur **F5** et définissez le paramètre de contexte devant être défini par l'utilisateur. Par exemple, le champ **Host** a pour valeur de paramètre `context.host`, dont le nom de paramètre est *host* dans le fichier de contexte. Sa valeur actuelle est *talend-dbms*.

- Puis dans le champ **Schema Type**, renseignez les informations sur le schéma. Si le schéma est stocké dans le noeud **Metadata** du **Repository** sous forme de métadonnée, sélectionnez-le après avoir sélectionné l'option **Repository**.
- Dans le champ **Query**, saisissez la requête SQL devant être exécutée sur la table spécifiée. Dans ce Job, effectuez un simple SELECT des colonnes de la table, qui seront ensuite affichées dans l'onglet **Run** grâce au composant **tLogRow**.
- Enfin, appuyez sur **F6** pour exécuter le Job.

```
Starting job tContextLoad at 16:19 05/03/2010.

tContextLoad_1 set key "host" with value "talend-dbms"
tContextLoad_1 set key "port" with value "3306"
tContextLoad_1 set key "database" with value "talend"
tContextLoad_1 set key "username" with value "root"
tContextLoad_1 set key "password" with value "toor"
12|4322 DP 76|BMW|purple|1|vaesneng|DFA6968
15|0142 CB 08|BMW|yellow|9|mauant|GCY8927
18|8545 GP 25|Mercedes|pink|2|vaesot|BNC6696
24|5382 KC 94|Volkswagen|grey|10|otbouh|GZE3017
40|8386 GH 71|Mercedes|orange|8|carmau|
67|0261 PY 55|Honda|green|6|hirtken|QXK4682
70|1850 CL 46|Mercedes|purple|7|oinegall|WAS9249
77|5653 HW 34|Toyota|yellow|2|carbone|GLN1029
91|6016 DF 08|Renault|pink|5|nanhirt|HWR9250
```

Les paramètres de contexte et les valeurs sélectionnés dans la table sont affichés dans la console **Run**.

tFixedFlowInput

Propriétés du tFixedFlowInput

Famille du composant	Misc	
Fonction	Le composant **tFixedFlowInput** génère autant de lignes et de champs que vous souhaitez alimenter par des valeurs fixes.	
Objectif	Le **tFixedFlowInput** permet de générer des données fixes à partir de variables internes.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et il est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Mode	Sélectionnez le mode, parmi les trois proposés, que vous souhaitez utiliser. **Use Single Table :** Saisissez la donnée que vous souhaitez générer dans la colonne Value correspondante. **Use Inline Table :** Ajoutez la ou les ligne(s) que vous souhaitez générer. **Use Inline Content :** Saisissez vos données à générer, séparées par les séparateurs préalablement définis dans les champs **Row** et **Field Separator**.
	Number of rows	Saisissez le nombre de lignes de données à générer.
	Values	Saisissez entre guillemets les valeurs correspondant aux colonnes définies dans la boîte de dialogue du schéma, via le bouton **Edit schema**.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé comme composant de début ou de milieu, il requiert donc un composant de sortie.	

Scénarios associés

Pour des scénarios associés, consultez :

- *Scénario 2 : Mettre les données de sortie en mémoire tampon du serveur d'application Web, page 1180.*
- *Scénario : Itérer une table de base de données et lister le nom des colonnes de la table, page 466.*

tMemorizeRows

Propriétés du tMemorizeRows

Famille de composant	Misc	
Fonction	Le composant **tMemorizeRows** mémorise temporairement une collection de données entrantes en séquence ligne par ligne et instancie cette collection en indexant chaque ligne mémorisée à partir de 0. Le nombre maximum de lignes à mémoriser à un moment donné est défini dans la vue **Basic settings**.	
Objectif	Le **tMemorizeRows** mémorise une séquence de lignes passant dans ce composant et permet au(x) composant(s) suivant(s) d'effectuer des opérations sur les lignes mémorisées, selon vos besoins.	
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (**Built-in**) soit distant dans le **Repository**. - Cliquez sur **Edit Schema** afin d'apporter des modifications au schéma. Notez que, si vous modifiez le schéma, il devient automatiquement **Built-In**. - Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le **Repository**. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Row count to memorize	Saisissez le nom de lignes à définir.
	Columns to memorize	Sélectionnez les colonnes à mémoriser du schéma d'entrée.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en tant qu'étape intermédiaire dans un flux de données ou comme dernière étape avant de lancer un sous-job.	

Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main. **Trigger** : Run if, On Component Ok, On Component Error. Liens d'entrée (d'un composant à un autre) : **Row** : Main. Pour plus d'informations concernant les connexions, consultez *Connection types* dans le **Guide Utilisateur** Talend Open Studio.

Scénario : Compter le nombre d'occurrences de différents âges

Ce scénario a pour objectif de compter le nombre d'âges différents se trouvant dans un groupe de 12 clients. Dans ce scénario, les données client sont générées de manière aléatoire.

Ce Job utilise les cinq composants suivants :

- **tRowGenerator** : ce composant génère 12 lignes de données client contenant les IDs, noms et âges des 12 clients.
- **tSortRow** : ce composant trie les 12 lignes selon les données de l'âge.
- **tMemorizeRows** : il mémorise temporairement un nombre spécifique de lignes de données entrantes à un moment donné et indexe les lignes de données mémorisées.
- **tJavaFlex** : ce composant compare la valeur des âges dans les données mémorisées par le composant précédent, compte le nombre d'occurrences des différents âges et affiche ces âges dans la vue **Run**.
- **tJava** : il affiche le nombre d'occurrences des différents âges.

Pour reproduire le scénario, procédez comme suit :

- Déposez les composants suivants de la **Palette** dans l'espace de modélisation graphique : **tRowGenerator**, **tSortRow**, **tMemorizeRows**, **tJavaFlex** et **tJava**.
- Cliquez-droit sur le **tRowGenerator**
  Dans le menu contextuel, sélectionnez le lien **Row** > **Main**.
- Cliquez sur le **tSortRow** afin de lier ces deux composants.

- Répétez l'opération pour connecter le **tSortRow**, le **tMemorizeRows** et le **tJavaFlex** à l'aide d'un lien **Row** > **Main**.
- Cliquez-droit sur le **tRowGenerator**
  Dans le menu contextuel, sélectionnez le lien **Trigger** > **OnSubjobOk**.
- Cliquez sur le **tJava** pour relier les composants.
- Double-cliquez sur le composant **tRowGenerator** afin d'ouvrir son éditeur.

- Dans cet éditeur, cliquez trois fois sur le bouton **[+]** pour ajouter trois colonnes, que vous nommerez : *id*, *name*, *age*.
- Dans la colonne **Type**, sélectionnez **Integer** pour les colonnes *id* et *age*.
- Dans la colonne **Length**, saisissez *50* pour la colonne *name*.
- Dans la colonne **Functions**, sélectionnez **random** pour *id* et *age*, puis sélectionnez **getFirstName** pour *name*.
- Dans le champ **Number of Rows for RowGenerator**, saisissez *12*.
- Dans la colonne **Column**, cliquez sur *age* afin d'ouvrir sa vue **Function parameters** dans la partie inférieure de l'éditeur.

- Dans la colonne **Value** de la vue **Function parameters**, saisissez l'âge minimum et l'âge maximum à générer pour les 12 clients. Dans cet exemple, saisissez respectivement *10* et *25*.
- Cliquez sur **OK** afin de sauvegarder la configuration.
- Dans la boîte de dialogue qui s'ouvre, cliquez sur **Yes** (Oui) afin de propager les modifications aux autres composants.
- Double-cliquez sur le **tSortRow** afin d'ouvrir sa vue **Component**.

- Dans le tableau **Criteria**, cliquez sur le bouton **[+]** afin d'ajouter une ligne.

- Dans la colonne **Schema column**, sélectionnez la colonne de données sur laquelle vous souhaitez baser le tri. Dans cet exemple, sélectionnez *age*, puisque les âges seront comparés et comptés.
- Dans la colonne **Sort num or alpha**, sélectionnez le type de tri. Dans cet exemple, comme *age* est un entier (integer), sélectionnez **num**, pour numérique.
- Dans la colonne **Order asc or desc**, sélectionnez **desc** comme ordre de tri.
- Double-cliquez sur le composant **tMemorizeRows** afin d'ouvrir sa vue **Component**.

- Dans le champ **Row count to memorize**, saisissez le nombre maximum de lignes à mémoriser à un moment donné. Puisque vous devez comparer les âges de deux clients à chaque fois, saisissez *2*.
  Ce composant mémorise deux lignes au maximum à un moment donné et indexe toujours la nouvelle ligne entrante comme 0 et la ligne précédente comme 1.
- Dans la colonne **Memorize** du tableau **Columns to memorize**, cochez la (les) case(s) de la (des) colonne(s) à mémoriser. Dans cet exemple, cochez la case correspondant à la colonne *age*.
- Double-cliquez sur le **tJavaFlex** afin d'ouvrir sa vue **Component**.

- Dans la zone **Start code**, saisissez le code Java qui sera appelé durant la phase d'initialisation. Dans cet exemple, saisissez `int count=0;` afin de déclarer une variable *count* et de lui assigner la valeur *0*.
- Dans la zone **Main code**, saisissez le code Java à appliquer à chaque ligne du flux de données. Dans ce scénario, saisissez :
```
if(age_tMemorizeRows_1[1]!=age_tMemorizeRows_1[0]) {
count++;
}
System.out.println(age_tMemorizeRows_1[0]);
```
Ce code compare les deux âges mémorisés à chaque fois par le **tMemorizeRows** et compte un changement à chaque fois que des âges différents sont trouvés. Ce code affiche également les âges indexés comme 0 par le **tMemorizeRows**.
- Dans la zone **End code**, saisissez le code qui sera appelé durant la phase de fermeture. Dans cet exemple, saisissez `globalMap.put("count", count);` afin d'écrire le résultat du compte.
- Double-cliquez sur le **tJava** afin d'ouvrir sa vue **Component**.

- Dans la zone **Code**, saisissez le code `System.out.println("Different ages : "+globalMap.get("count"));` afin de récupérer le résultat du compte.
- Appuyez sur **F6** afin d'exécuter le Job. Le résultat s'affiche dans la console de la vue **Run**.

```
Starting job test at 18:32 23/09/2010.

[statistics] connecting to socket on port 3348
[statistics] connected
25
24
19
18
17
16
14
13
13
13
12
11
Different ages: 10
[statistics] disconnected
Job test ended at 18:32 23/09/2010. [exit code=0]
```

Dans la console, vous pouvez constater qu'il y a dix âges différents dans le groupe des 12 clients.

tMsgBox

Propriétés du tMsgBox

Famille de composant	Misc	
Fonction	Le composant **tMsgBox** ouvre une boîte de dialogue contenant un bouton **OK**, nécessitant une action de la part de l'utilisateur.	
Objectif	Ce composant est une pause graphique dans le cours du traitement.	
Basic settings	*Title*	Le texte saisi s'affiche dans la barre de titre de la boîte de dialogue.
	Buttons	Liste des boutons que vous souhaitez inclure dans la boîte de dialogue. Les combinaisons de bouton sont restreintes et ne peuvent être changées.
	Icon	Icône de la barre de titre de la boîte de dialogue.
	Message	Texte libre à afficher dans la boîte de dialogue. Le texte peut être dynamique (ex : reprendre un nom de fichier).
Utilisation	Ce composant peut être utilisé comme étape intermédiaire du traitement ou comme composant de début ou de fin de Job. Il peut être connecté à l'aide d'un lien **Row** ou **Iterate**.	
Limitation	Si vous utilisez Perl, assurez-vous que le package adéquat est installé sur votre ordinateur.	

Scénario : Test de type 'Hello world!'

Le scénario suivant représente un Job à un seul composant utilisé pour afficher l'heure et la date à la place du message "Hello World!".

- Cliquez et déposez un composant **tMsgBox** dans l'espace de modélisation.
- Paramétrez les propriétés d'affichage du message :

- 'Title' est le titre de la boîte de message, vous pouvez utiliser n'importe quelle variable.

- Dans le champ **Message**, saisissez le texte `"Current date is: "` entre guillemets doubles concaténé à l'aide d'un +, puis appuyez sur **Ctrl+Espace** afin d'afficher la liste d'auto-complétion, et sélectionnez la routine système `TalendDate.getCurrentDate`. Entourez-la de parenthèses.
- Cliquez sur l'onglet **Run** puis exécutez le Job.

Le message affiche le texte défini précédemment et requiert une action de l'utilisateur pour disparaître et passer au composant suivant ou terminer le Job.

Après avoir cliqué sur le bouton **OK**, le log de la vue **Run** est mis à jour.

Voir également : *Exécution d'un job* du **Guide utilisateur** de **Talend Open Studio**.

tRowGenerator

Propriétés du tRowGenerator

Famille de composant	Misc	
Fonction	Le composant **tRowGenerator** génère autant de lignes et de champs que vous souhaitez alimenter par des valeurs prises de façon aléatoire dans une liste définie.	
Objectif	Ce composant peut être utilisé à des fins de test, pour créer un flux d'entrée dans le cadre de jeux de tests.	
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé pour ce composant uniquement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisable dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
Basic settings	*RowGenerator editor*	L'éditeur vous permet de définir précisément les colonnes et la nature des données à générer. Vous pouvez prédéfinir des routines ou saisir vous-même la fonction à utiliser pour générer les données souhaitées.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	La simplicité d'utilisation de l'éditeur du **tRowGenerator** vous permet de générer des données de façon aléatoire sans connaissance des langages Perl et Java.	
Limitation	n/a	

L'éditeur du **tRowGenerator** s'ouvre dans une fenêtre séparée constituée de deux parties :

- dans la partie supérieure de la fenêtre, une vue de définition de **Schema**,
- dans la partie inférieure, une vue Function de définition et d'aperçu des fonctions.

Définition du schéma

Pour commencer, vous devez définir la structure des données à générer.

- Ajoutez autant de colonnes que nécessaire à l'aide du bouton **plus [+]**.
- Dans la zone **Columns**, saisissez les noms des colonnes à créer et cochez la case **Key** si nécessaire.

- Assurez-vous d'avoir correctement défini la nature des données contenues dans la colonne en sélectionnant le type dans la colonne **Type**. En fonction du type sélectionné, la liste des fonctions proposées dans la colonne **Function** diffère. Cette information est obligatoire.

- D'autres informations, bien que facultatives, peuvent être utiles, telles que les colonnes **Length**, **Precision** ou **Comment**. Mais vous avez la possibilité de cacher les colonnes que vous n'utilisez pas, en cliquant sur le bouton **Columns** dans la barre d'outils et en décochant les colonnes que vous souhaitez cacher.
- Dans la zone **Function**, sélectionnez la routine/fonction prédéfinie si l'une des propositions correspond à vos besoins. Vous pouvez également ajouter à cette liste toute routine que vous avez développée et stockée dans la partie **Routine** du **Repository**. Vous avez enfin la possiblité de saisir directement la fonction que vous souhaitez utiliser dans la vue **Function**. Voir également : *Définition d'une fonction, page 1205*.
- Cliquez sur **Refresh** pour visualiser un aperçu des données générées.
- Saisissez le nombre de lignes à générer. Plus le nombre de lignes est élevé, plus la durée d'exécution sera longue pour cette opération de génération.

La liste des fonctions proposée diffère selon la langue de votre projet (Perl ou Java).

Définition d'une fonction

Vous avez sélectionné les trois points **[...]** dans la colonne **Function** de l'éditeur du Schéma, car vous souhaitez personnaliser la fonction de génération.

- Sélectionnez l'onglet **Function parameters**.
- La zone **Parameter** affiche **Customized parameter** comme nom du paramètre (en lecture seule).

- Dans la zone **Value**, saisissez la fonction Perl ou Java que vous souhaitez utiliser pour générer des données.
- Cliquez sur l'onglet **Preview** et cliquez sur le bouton **Preview** pour visualiser un échantillon des données qui seront générées.

Scénario : Génération aléatoire de fichiers de test

Le scénario suivant décrit un Job de deux composants, générant 50 lignes composées de la manière suivante : un ID aléatoire compris entre 1 et 3, un nom et un prénom pris de manière aléatoire dans une liste de chaînes de caractères, et le champ Date définie dans une période donnée.

- Cliquez et déposez un composant **tRowGenerator** et un composant **tLogRow** dans l'espace de modélisation.
- Cliquez-droit sur le composant **tRowGenerator** et sélectionnez le lien **Row** > **Main**. Faites glisser ce lien jusqu'au composant **tLogRow**.
- Double-cliquez sur le composant **tRowGenerator** pour ouvrir l'éditeur.
- Définissez les champs à générer.

- La colonne ID (*random_ID*) est de type entier (**integer**), les noms et prénoms sont de type chaîne de caractères (**String**) et la *Date* est de type **Date**.
- Dans la colonne **Function**, sélectionnez la fonction adaptée dans la liste ou les trois points **[...]** pour la fonction personnalisée.
- Dans l'onglet **Function parameters**, définissez les valeurs à prendre de façon aléatoire.

Parameter	Value	Comment
customize parameter	1,2,3	

- Les colonnes *First_Name* et *Last_Name* sont à générer à l'aide d'une fonction `getAsciiRandomString` prédéfinie dans les routines système. Par défaut, la longueur définie est de 6 caractères, mais vous pouvez la changer.

- La colonne *Date* appelle également une fonction prédéfinie `getRandomDate`. Vous pouvez éditer les valeurs de paramètre dans l'onglet **Function parameters**.
- Dans le champ **Number of Rows**, paramétrez le nombre de lignes à 50.
- Cliquez sur **OK** pour valider les paramètres.
- Double-cliquez sur le composant tLogRow pour visualiser les paramètres dans l'onglet **Basic settings**. Retenez les paramètres par défaut.
- Appuyez sur **F6** pour exécuter le Job.

```
Starting job JavaRowGenerate at 11:44 10/04/2007.
Running process with context: Default
1|iH6mtj|y3hSXH|Wed Apr 16 11:44:15 CEST 2008
3|cityeQ|uvnKkO|Sun Jan 06 11:44:16 CET 2008
2|X2OODP|1SVzKT|Wed Oct 26 11:44:16 CEST 2005
1|DSLuE1|S8u15i|Sat Mar 04 11:44:16 CET 2006
3|cc4znX|yuc9cf|Thu Jan 20 11:44:16 CET 2005
3|NwK3PN|lnNyDU|Mon Jun 06 11:44:16 CEST 2005
3|CtO6Ba|pCQgwp|Sat Aug 07 11:44:16 CEST 2004
3|XMt7KN|SUIzFn|Sun May 09 11:44:16 CEST 2004
1|jiPor7|l45rp7|Wed Aug 25 11:44:16 CEST 2004
1|OBN4TX|hywNdP|Sat Feb 09 11:44:16 CET 2008
2|kbVUuE|s21FA0|Tue Jan 27 11:44:16 CET 2004
2|zQlteU|VNHb5w|Sun Oct 05 11:44:16 CEST 2008
3|4nrniu|5Tbnxd|Fri Nov 14 11:44:16 CET 2008
2|7A0qqu|s3jEzJ|Tue Feb 13 11:44:16 CET 2007
3|FgVoTg|uYhJmF|Thu Mar 01 11:44:16 CET 2007
1|Sp4e9P|zXcTK5|Thu Apr 08 11:44:16 CEST 2004
1|E3A4Q3|vKNocj|Wed Jan 11 11:44:16 CET 2006
1|pOMGpG|koklW2|Thu Apr 22 11:44:16 CEST 2004
3|9XOgm8|CCbFya|Sat Feb 18 11:44:16 CET 2006
2|itZXWx|LhQBx1|Tue Apr 18 11:44:16 CEST 2006
1|On1dwv|E7Yaqn|Fri Dec 10 11:44:16 CET 2004
1|UPTZ9M|8s902c|Fri Nov 11 11:44:16 CET 2005
1|WhIT0u|KWn8hO|Mon May 07 11:44:16 CEST 2007
1|LbbeFy|Evn1lc|Thu Mar 16 11:44:16 CET 2006
```

Les 50 lignes sont générées selon les paramètres définis dans l'éditeur du composant **tRowGenerator** et sont affichées en sortie standard de la vue **Run**.

Composants Orchestration

Ce chapitre passe en revue les propriétés des principaux composants présents dans la famille **Orchestration** de la **Palette** de **Talend Open Studio**.

La famille **Orchestration** regroupe les composants vous permettant de séquencer et d'orchestrer les tâches ou les traitements de vos Jobs et sous-Jobs.

tFileList

Le **tFileList** appartient à deux familles : File et Orchestration. Pour plus d'informations concernant le composant **tFileList**, consultez *tFileList, page 970.*.

tFlowToIterate

Propriétés du tFlowToIterate

Famille du composant	Orchestration	
Fonction	Le composant **tFlowToIterate** transforme un flux de données en une liste.	
Objectif	Le **tFlowToIterate** permet de transformer un flux traitable en données non traitables.	
Basic settings	*Use the default (key, value) in global variables*	Lorsque cette case est cochée, le système utilise la valeur par défaut de la variable gobale utilisée dans le Job.
	Customize	**key** : Entrez le nom de la nouvelle variable globale. Appuyez sur **Ctrl+Espace** pour accéder à l'ensemble des variables disponibles, qu'elles soient globales ou définies par l'utilisateur.
		value : Cliquez dans la cellule pour accéder à la liste des colonnes rattachées à la variable globale définie.
Utilisation	Le **tFlowToIterate** n'est pas un composant de début, et il requiert un composant de sortie.	
Global Variables		**Number of Lines** : indique le nombre de lignes traitées. Disponible comme variable **After**. Retourne un entier. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Iterate. **Trigger** : Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Main. Pour plus d'informations concernant les liens, consultez *Types de connexions*, dans le **Guide Utilisateur Talend Open Studio**.

Scénario : Transformer un flux de données en liste

Le scénario suivant décrit un Job permettant de lire une liste de fichiers à partir d'un fichier d'entrée donné. Il effectue également une itération sur chacun des fichiers, sélectionne les données d'entrée et affiche le résultat en sortie dans la console de log **Run**.

- A partir de la **Palette**, cliquez-déposez les composants suivants dans l'espace de modélisation : deux **tFileInputDelimited**, un **tFlowToIterate** et un **tLogRow**.
- Cliquez-droit sur chacun des composants pour les relier entre eux. Connectez le premier **tFileInputDelimited** au **tFlowToIterate** à l'aide d'un lien de type **Row Main**, puis le **tFlowToIterate** au deuxième **tFileInputDelimited** à l'aide d'un lien de type **Iterate**, enfin le second **tFileInputDelimited** au **tLogRow** à l'aide d'un lien de type **Row Main**.

- Dans l'espace graphique, sélectionnez le premier **tFileInputDelimited**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du composant **tFileInputDelimited**.
- Dans la vue **Basic settings**, cliquez sur le bouton **[...]** situé à côté du champ **File Name** pour déterminer le chemin d'accès au fichier d'entrée.

Le champ **File Name** est obligatoire.

Le fichier d'entrée utilisé dans ce scénario s'appelle *Customers*. Il s'agit d'un fichier texte contenant trois autres fichiers texte simples : *Name*, *E-mail* et *Address*. Le fichier *Name* est constitué d'une colonne contenant le nom des clients et le fichier *E-mail* d'une colonne contenant l'adresse email de ces clients. Le troisième fichier texte, *Address*, est composé d'une colonne comprenant l'adresse postale des clients.

- Renseignez les autres champs en fonction de votre fichier. Pour plus d'informations sur le tFileInputDelimited, consultez les *Propriétés du tFileInputDelimited, page 915*. Dans ce scénario, l'en-tête, le pied de page et le nombre limite de lignes à traiter ne sont pas définis. Donc, laissez les champs **Header** (en-tête) et **Footer** (pied de page) et **Limit** par défaut.

- Cliquez sur le bouton **Edit schema** pour décrire la structure des données du fichier d'entrée. Dans ce scénario, le schéma comporte une seule colonne appelée *FileName*.

- Dans l'espace graphique, sélectionnez le **tFlowToIterate**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du composant **tFlowToIterate**.

- Cochez la case **Use the default (key, value) in global variables** si besoin. Cette option permet d'utiliser la valeur par défaut de la variable globale.
- Cliquez sur le bouton **[+]** pour ajouter une ligne de paramètres puis définissez vos variables.
- Cliquez dans la cellule de la colonne **key** pour modifier le nom de la variable selon vos désirs.

Vous pouvez appuyer sur **Ctrl+Espace** à partir de la cellule de la colonne **key** pour accéder à la liste des variables globales et des variables utilisateur.

- Dans l'espace graphique, sélectionnez le second composant **tFileInputDelimited**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) de ce second **tFileInputDelimited**.

- Dans le champ **File Name**, renseignez le nom du fichier en utilisant la variable appropriée. Vous devez utiliser la syntaxe correspondant au langage employé (Perl ou Java). En Perl, la syntaxe exacte est *.$_globals{tFlowToIterate}{'Name_of_File}*. En Java, la syntaxe correcte est *+globalMap.get("file")*.
- Remplissez les autres champs selon vos besoins. Pour plus d'informations, consultez les *Propriétés du tFileInputDelimited, page 915*.
- Dans l'espace graphique, sélectionnez le dernier composant, le **tLogRow**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tLogRow**.

- Définissez les paramètres selon vos besoins. Pour plus d'informations, consultez *Propriétés du tLogRow, page 1163*.
- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

```
Starting job flow_to_iterate at 16:32 25/07/2008.
col1: Madison Moore
col1: Andrew Taylor
col1: Christopher Anderson
col1: madison_moor@hotmail.com
col1: Andrew_taylor@usamail1.com
col1: Madison Moore
col1:     100 MAIN ST
col1:     PO BOX 1022
col1:     SEATTLE WA 98104
col1:     USA
col1: Andrew Taylor
col1:     300 BOYLSTON AVE E
col1:     SEATTLE WA 98102
col1:     USA
Job flow_to_iterate ended at 16:32 25/07/2008. [exit code=0]
```

La console affiche, pour chaque ligne, le nom du client, son e-mail et son adresse, précédés du nom de la colonne du schéma.

tForeach

Propriétés du tForeach

Famille de composant	Orchestration	
Fonction	Le composant **tForeach** crée une boucle sur une liste pour un lien **Iterate**.	
Objectif	Le **tForeach** vous permet de créer une boucle sur une liste pour un lien **Iterate**.	
Basic settings	*Values*	Utilisez le bouton **[+]** pour ajouter des lignes au tableau **Values**. Cliquez ensuite sur les champs afin de saisir, entre guillemets, les valeurs de la liste sur laquelle effectuer une boucle.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Le **tForeach** est un composant d'entrée. Un lien **Iterate** est requis pour relier ce composant à un autre.	
Limitation	n/a	

Scénario : Faire une boucle sur une liste et récupérer les valeurs

Ce scénario décrit un Job à deux composants dans lequel une liste est créée et sur laquelle est effectuée une boucle, via un composant **tForeach**. Les valeurs sont récupérées par un composant **tJava**.

- Déposez de la **Palette** dans l'espace de modélisation graphique les composants **tForeach** et **tJava**.

- Reliez le **tForeach** au **tJava** à l'aide d'un lien **Row > Iterate**.
- Double-cliquez sur le **tForeach** afin d'ouvrir sa vue **Basic settings**.

- Cliquez sur le bouton pour ajouter autant de lignes que nécessaire dans le tableau **Values**.
- Cliquez sur les champs **Value** afin de saisir les valeurs de la liste, entre guillemets doubles.
- Double-cliquez sur le **tJava** afin d'ouvrir sa vue **Basic settings**.

- Saisissez le code Java suivant dans la zone **Code** :
  ```
  System.out.println(globalMap.get("tForeach_1_CURRENT_VALUE")
  +"_out");
  ```
- Sauvegardez le Job et appuyez sur **F6** pour l'exécuter.
  La console affiche la liste des valeurs récupérées du **tForeach**, chaque valeur ayant le suffixe _out.

Job
Con
Com
Run
Pro
Mod
Tale
Sch
Job
Job tForEach_2
Scheduler
Basic Run
Debug Run
Advanced Settings
Target Exec
Execution
Run
Kill
Clear
Starting job tForEach_2 at 14:28 04/10/2010.
[statistics] connecting to socket on port 3409
[statistics] connected
element_1_out
element_2_out
element_3_out
element_4_out
[statistics] disconnected
Job tForEach_2 ended at 14:28 04/10/2010. [exit code=0]
Line limit
100

tInfiniteLoop

Propriétés du tInfiniteLoop

Famille de composant	Orchestration	
Fonction	Le composant **tInfiniteLoop** fait une boucle à l'infini sur l'exécution d'une tâche.	
Objectif	Le **tInfiniteLoop** permet d'exécuter automatiquement une tâche ou un Job dans une boucle, à l'infini.	
Basic settings	*Wait at each iteration (in seconds)*	Saisissez le temps, en secondes, d'attente entre chaque itération.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Le composant **tInifniteLoop** est un composant de début et requiert une connexion de type Iterate au composant suivant.	
Global Variables		**Current iteration** : indique le nombre d'itérations. Disponible comme variable **Flow**. Retourne un entier. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur** Talend Open Studio.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Main. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** Talend Open Studio.
Limitation	n/a	

Scénario associé

Pour un exemple d'utilisation proche, consultez le *Scénario : Exécuter un Job à partir d'une boucle, page 1224* du composant tLoop.

tIterateToFlow

Propriétés du tIterateToFlow

Famille de composant	Orchestration	
Fonction	Le composant **tIterateToFlow** transforme une liste en flux de données pouvant être traité.	
Objectif	Ce composant permet de transformer des données non traitables en flux traitable.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Lors de l'utilisation du **tIterateToFlow**, le schéma doit être défini.
		Built-in : Le schéma est créé et conservé pour ce composant uniquement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Mapping	**Column** :Saisissez le nom des colonnes à créer. **Value** : Appuyez sur **Ctrl+Espace** pour accéder à toutes les variables disponibles soit globales soit celles que vous avez définies.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant n'est pas un composant de début (fond vert) et requiert un composant de sortie.	
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main. **Trigger** : Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.

Scénario : Transformer une liste de fichiers en flux de données

Le scénario suivant décrit un Job permettant de faire une boucle sur une liste de fichiers, de récupérer leur nom, ainsi que la date en cours, pour ensuite transformer ces données en un flux qui sera affiché dans la console.

- Cliquez et déposez les composants suivants : **tFileList**, **tIterateToFlow** et **tLogRow**.
- Connectez le **tFileList** au **tIterateToFlow** à l'aide d'un lien **Iterate** et connectez le Job au composant **tLogRow** à l'aide d'un lien **Row main**.
- Dans la vue **Component** du composant **tFileList**, donnez le chemin d'accès au répertoire contenant la liste de fichiers.

- Dans cet exemple, les fichiers sont trois fichiers .txt simples stockés dans le répertoire *Countries*.
- Dans la liste **Case sensitive**, sélectionnez **No** (Non) pour ne pas tenir compte de la casse.
- Sélectionnez le composant **tIterateToFlow** et cliquez sur **Edit Schema** pour modifier le nouveau schéma.

- Ajoutez deux nouvelles colonnes : *Filename* de type **String** et *Date* de type **date**. Assurez-vous d'avoir paramétré le bon modèle en Java.
- Cliquez sur **OK** pour valider.

- Une boîte de dialogue s'ouvre et vous demande si vous souhaitez propager les changements. Cliquez sur **Yes** (Oui).
- Notez que ce nouveau schéma apparaît dans le tableau **Mapping**.

- Dans chaque cellule de la colonne **Value**, appuyez sur **Ctrl+Espace** pour accéder à la liste des variables globales et à celles que vous avez définies.
- Pour la colonne *Filename*, utilisez la variable globale : `tFileList_1CURRENT_FILEPATH`. Elle reprend le chemin d'accès courant pour recueillir le nom de chaque fichier, le Job fait une boucle sur ce répertoire.
- Pour le colonne *Date*, utilisez la routine Talend : `Date.GetDate` (en Perl) ou `TalendDate.getCurrentDate()` (en Java)
- Puis dans la vue **Component** du composant **tLogRow**, cochez la case **Print values in cells of a table**.
- Enregistrez votre Job et exécutez-le, en appuyant sur **F6**.

```
Starting job tIterateToFlow at 10:28 10/03/2010.

[statistics] connecting to socket on port 3773
[statistics] connected
.----------------------------+-------------------.
|                   tLogRow_1                    |
|=---------------------------+-------------------=|
|Filename                    |Date               |
|=---------------------------+-------------------=|
|D:\Input\Countries\in-01.txt|2010-03-10 10:28:56|
|D:\Input\Countries\in-02.txt|2010-03-10 10:28:56|
|D:\Input\Countries\in-03.txt|2010-03-10 10:28:56|
'----------------------------+-------------------'

[statistics] disconnected
Job tIterateToFlow ended at 10:28 10/03/2010. [exit code=0]
```

Le chemin d'accès apparaît dans la colonne *Filename* et la date en cours apparaît dans la colonne *Date*.

tLoop

Propriétés du tLoop

Famille de composant	Orchestration	
Fonction	Le composant **tLoop** fait une boucle sur l'exécution d'une tâche.	
Objectif	Le **tLoop** permet d'exécuter automatiquement une tâche ou un Job dans une boucle.	
Basic settings	*Loop Type*	Sélectionnez le type de boucle à effectuer : **For** ou **While**. **For** : La tâche ou le Job est exécuté(e) pour le nombre de boucle défini. **While** : La tâche ou le Job est exécuté(e) jusqu'à ce que la condition soit rencontrée.
For	*From*	Saisissez le numéro de la première instance sur laquelle la boucle doit commencer. Si vous sélectionnez l'instance numéro 2 comme instance de départ et que vous prenez un écart de 2, cela signifie que la boucle se fera sur tous les nombres pairs.
	To	Saisissez la dernière instance sur laquelle la boucle doit passer.
	Step	Saisissez l'écart entre deux instances avec lequel la boucle doit s'effectuer. Un écart de 2 signifie que la boucle passe toutes les deux instances.
While	*Declaration*	Saisissez une expression déclenchant la boucle.
	Condition	Saisissez la condition devant être rencontrée pour que la boucle s'arrête.
	Iteration	Saisissez l'expression décrivant l'opération à effectuer à chaque boucle.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Le composant **tLoop** est un composant de début et requiert une connexion de type Iterate au composant suivant.	
Global Variables		**Current value** : indique la valeur actuelle. Disponible comme variable **Flow**. Retourne un entier. **Current iteration** : indique le nombre d'itérations. Disponible comme variable **Flow**. Retourne un entier. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur** **Talend Open Studio**.

Connections		Liens de sortie (d'un composant à un autre) : **Row** : Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** Talend Open Studio.
Limitation	n/a	

Scénario : Exécuter un Job à partir d'une boucle

Ce scénario décrit un Job constitué d'un Job parent et d'un Job enfant. Le Job parent exécute une boucle qui permet d'exécuter un Job enfant n nombre de fois, avec une pause entre chaque exécution.

- Dans le Job parent, cliquez et déposez les composants **tLoop**, **tRunJob** et **tSleep** dans l'espace de modélisation.
- Connectez le composant **tLoop** au **tRunJob** à l'aide d'une connexion de type **Iterate**.
- Puis connectez le **tRunJob** au **tSleep** à l'aide d'une connexion de type **Row**.
- Dans le Job enfant, cliquez et déposez les composants suivants : **tPOP**, **tFileInputMail** et **tLogRow**.
- Dans l'onglet **Basic settings** du composant **tLoop**, saisissez le numéro de l'instance sur lequel commencer (*1*), le numéro de l'instance sur lequel finir (*5*) et l'écart à utiliser (*1*).

- Dans l'onglet **Basic settings** du composant **tRunJob**, sélectionnez le Job enfant dans la liste des Jobs proposés, pour cet exemple : *popinputmail*.
- Sélectionnez le contexte, si nécessaire. Pour cet exemple, le contexte est celui par défaut (*default*) ne contenant aucune variable.
- Dans l'onglet **Basic settings** du composant **tSleep**, saisissez la durée de la pause en secondes dans le champ **Pause**. Pour cet exemple, la pause est de *3 seconds*.
- Dans l'onglet **Basic settings** du Job enfant, définissez les paramètres de connexion au serveur pop.
- Dans l'onglet **Basic settings** du composant **tFileInputMail**, sélectionnez une variable globale dans le champ **File Name**, pour collecter le fichier courant dans le répertoire défini dans le composant **tPOP**. Appuyez sur **Ctrl+Espace** pour accéder à la liste des variables. Pour cet exemple, la variable à utiliser est :
  `$_globals{tPOP_1}{CURRENT_FILEPATH}` (pour Perl)
  `((String)globalMap.get("tPOP_1_CURRENT_FILEPATH"))` (pour Java)
- Définissez le **Schema** pour qu'il inclue l'élément du mail à traiter. Les champs du schéma sont les suivants : *author*, *topic*, *delivery date* et *number of lines*.
- Dans la colonne **Mail part** de la table **Mail Parts**, saisissez le champ de l'e-mail correspondant à chaque colonne définie dans le schéma. Par exemple : *author* provient du champ *From* du fichier e-mail.
- Puis connectez les composants **tFileInputMail** et **tLogRow**.
- Appuyez sur **F6** pour exécuter le Job, et vérifiez les résultats de l'exécution dans la vue **Run**.

tPostjob

Propriétés du tPostjob

Famille du composant	Orchestration	
Fonction	Le composant **tPostjob** démarre l'exécution d'un post-job.	
Objectif	Le **tPostjob** déclenche une action requise après l'exécution d'un Job.	
Utilisation	Le **tPostjob** est un composant de début. Il ne peut être relié au composant suivant qu'avec une connexion de type **Iterate**.	
Connections		Liens de sortie (d'un composant à un autre) : **Trigger** : OnComponentOk. Liens d'entrée (d'un composant à un autre) : **Trigger** : Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	n/a	

Pour plus d'informations sur le composant **tPostjob**, consultez la section *Utiliser les composants tPrejob et tPostjob* du **Guide Utilisateur** de **Talend Open Studio**.

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tPrejob

Propriétés du tPrejob

Famille du composant	Orchestration	
Fonction	Le composant **tPrejob** démarre l'exécution d'un pré-job.	
Objectif	Le **tPrejob** déclenche une action requise pour l'exécution d'un Job.	
Utilisation	Le **tPrejob** est un composant de début. Il ne peut être relié au composant suivant qu'avec une connexion de type **Iterate**.	
Connections		Liens de sortie (d'un composant à un autre) : **Trigger** : OnComponentOk. Liens d'entrée (d'un composant à un autre) : **Trigger** : Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** Talend Open Studio.
Limitation	n/a	

Pour plus d'informations sur le composant **tPrejob**, consultez la section *Utiliser les composants tPrejob et tPostjob* du **Guide Utilisateur** de Talend Open Studio.

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tReplicate

Propriétés du tReplicate

Famille de composant	Orchestration	
Fonction	Le composant **tReplicate** duplique le schéma entrant en deux flux identiques.	
Objectif	Ce composant permet d'effectuer différentes opérations sur le même schéma.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé pour ce composant uniquement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository: Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Utilisation	Ce composant n'est pas un composant de début (fond vert) puisqu'il requiert un composant d'entrée et un composant de sortie.	
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main **Trigger** : Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** :Main, Reject. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.

Scénario associé

Pour un exemple d'utilisation du tReplicate, consultez le *tReplaceList, page 163*

tRunJob

Le **tRunJob** appartient à deux familles : Orchestration et System. Pour plus d'informations concernant le composant **tRunJob**, consultez *tRunJob, page 1352*.

tSleep

Propriétés du tSleep

Famille de composant	Orchestration	
Fonction	Le composant **tSleep** effectue une pause lors de l'exécution d'un Job.	
Objectif	Ce composant permet d'identifier les goulots d'étranglement éventuels à l'aide d'une pause lors de l'exécution du Job à des fins de test ou de monitoring. En production, il peut être utilisé pour tout type de pause dans le traitement d'un Job, pour alimenter un flux entrant par exemple.	
Basic settings	*Pause (in second)*	Temps en seconde pendant lequel le Job est suspendu.
Utilisation	Le composant **tSleep** est généralement utilisé comme composant intermédiaire mettant un Job en pause, avant de le reprendre.	
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject, Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	n/a	

Scénario associé

Pour un exemple d'utilisation du **tSleep**, consultez le *Scénario : Exécuter un Job à partir d'une boucle, page 1224*.

tUnite

Propriétés du tUnite

Famille de composant	Orchestration	
Fonction	Le composant **tUnite** fusionne des données de diverses sources, basées sur un même schéma.	
Objectif	Ce composant centralise des données provenant de sources diverses et hétérogènes.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé pour ce composant uniquement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant n'est pas un composant de début; il requiert un ou plusieurs composants en entrée et un composant de sortie.	
Global Variables		**Number of lines** : indique le nombre de lignes traitées. Disponible comme variable **After**. Retourne un entier. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main. **Trigger** : Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.

Scénario : Itération sur des fichiers et fusion de contenu

Le Job suivant effectue une itération sur une liste de fichiers puis fusionne leur contenu et affiche le contenu final des deux colonnes dans la console.

- Cliquez et déposez les composants suivants dans l'espace de modélisation : **tFileList**, **tFileInputDelimited**, **tUnite** et **tLogRow**.
- Connectez le composant **tFileList** au **tFileInputDelimited** à l'aide d'un lien **Iterate** et connectez les autres composants à l'aide de liens **Row main**.
- Dans l'onglet **Basic settings** du composant **tFileList**, sélectionnez le répertoire dans lequel les fichiers à fusionner sont stockés.

- Dans la liste **Case Sensitive**, sélectionnez **Yes** (Oui) si vous souhaitez tenir compte de la casse, ou **No** (Non) si vous ne vous voulez pas la prendre en compte. Pour cet exemple, sélectionnez **Yes**.
- Les fichiers sont très basiques et contiennent une liste de pays et leur score respectif.

- Sélectionnez le composant **tFileInputDelimited** et affichez son onglet **Basic settings.**
- Dans cet exemple, les paramètres de connexion des fichiers d'entrée ne sont pas stockés dans les métadonnées du **Repository**. Ainsi, sélectionnez l'option **Built-In** dans le champ **Property type** et paramétrez les propriétés manuellement.

- Pour renseigner le champ **File Name**, appuyez sur **Ctrl+Espace** pour accéder à la liste des variables. Pour traiter tous les fichiers du répertoire défini dans le composant **tFileList**, sélectionnez `tFileList.CURRENT_FILEPATH` dans la liste des variables globales.
- Conservez les paramètres par défaut des champs **Row** et **Field separators**, ainsi que des autres champs.
- Cliquez sur le bouton **Edit Schema** et définissez manuellement le schéma à deux colonnes pour qu'il corresponde au contenu des fichiers d'entrée.

- Pour cet exemple, les deux colonnes sont *Country* et *Points*. Elles peuvent toutes les deux prendre une valeur nulle, ainsi cochez les cases de la colonne **Nullable**.
- La colonne *Country* sera de type String, et la colonne *Points* sera de type Integer.

- Cliquez sur **OK** pour valider les paramètres, puis acceptez la propagation du schéma dans le reste du Job.
- Puis sélectionnez le composant **tUnite** et affichez la vue **Component**. Notez que le schéma de sortie reflète exactement celui d'entrée et qu'il est en lecture seule.
- Dans la vue **Component** du composant **tLogRow**, cochez la case **Print values in cells of the table** pour afficher correctement les valeurs de sortie.
- Enregistrez le Job et exécutez-le.

```
Starting job fetch at 16:11 08/03/2010.

.-------+------.
|  tLogRow_1   |
|=------+-----=|
|Country|Points|
|=------+-----=|
|france |12    |
|usa    |10    |
|france |14    |
|uk     |13    |
|usa    |9     |
|france |5     |
|france |16    |
|uk     |13    |
'-------+------'
Job fetch ended at 16:11 08/03/2010. [exit code=0]
```

La console affiche les données des différents fichiers, fusionnées dans une seule table.

tWaitForFile

Propriétés du tWaitForFile

Famille de composant	Orchestration	
Fonction	Le composant **tWaitForFile** fait une boucle sur un répertoire donné à la recherche d'ajout ou de suppression de fichier puis déclenche un sous-Job lorsque la condition est rencontrée.	
Objectif	Ce composant permet de déclencher un Job en fonction d'une condition liée à la présence ou au retrait d'un fichier.	
Basic settings	*Wait at each iteration (in seconds)*	Paramétrez l'intervalle de temps entre chaque boucle.
	Max. iterations (infinite if empty)	Nombre de vérifications à effectuer avant que le Job ne s'arrête.
	Directory to scan	Nom du répertoire à vérifier.
	File mask	Masque du fichier recherché.
	Include subdirectories	Cochez cette case pour inclure les sous-dossiers.
	Case sensitive	Cochez cette case pour tenir compte de la casse.
	Include present file	Cochez cette case pour inclure le fichier courant.
	Trigger action when	Sélectionnez la condition à remplir pour effectuer l'action : **A file is created** : un fichier est créé. **A file is deleted** : un fichier est supprimé. **A file is updated** : un fichier est mis à jour **A file is created or updated or deleted** : un fichier est créé, supprimé ou mis à jour.
	Then	Sélectionnez l'action à effectuer : soit stopper l'itération lorsque la condition est rencontrée (**exit loop**) soit continuer la boucle jusqu'à ce que le nombre maximum d'itérations soit atteint (**continue loop**).
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé pour ce composant uniquement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.

		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisable dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Wait for file to be released*	Cochez cette case pour empêcher l'exécution du Job si votre fichier est ouvert.
Utilisation	Ce composant est un composant de début (ou de déclenchement) pour le sous-Job exécuté en fonction de la condition établie. Ainsi, ce composant doit être connecté à un sous-job via un lien Iterate.	
Global Variables		**Current iteration** : indique le nombre d'itérations. Disponible comme variable **Flow**. Retourne un entier. **Present File** : indique le nom du fichier courant dans l'itération ayant activé le déclenchement. Disponible comme variable **Flow**. Retourne une chaîne de caractères. **Deleted File** : indique le nom et le chemin d'accès du fichier supprimé qui a activé le déclenchement. Disponible comme variable **Flow**. Retourne une chaîne de caractères. **Created File Name** : indique le nom et le chemin d'accès du fichier nouvellement créé ayant activé le déclenchement. Disponible comme variable **Flow**. Retourne une chaîne de caractères. **Updated File** : indique le nom et le chemin d'accès du fichier mis à jour, activant ainsi le déclenchement. Disponible comme variable **Flow**. Retourne une chaîne de caractères. **File Name** : indique le nom d'un fichier créé, supprimé ou mis à jour, activant ainsi le déclenchement. Disponible comme variable **Flow**. Retourne une chaîne de caractères. **Not Updated File Name** : indique les noms des fichiers n'ayant pas été mis à jour, activant ainsi le déclenchement. Disponible comme variable **Flow**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur** **Talend Open Studio**.

Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : OnSubjobOk, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate. **Trigger** : OnSubjobOk, Run if, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** Talend Open Studio.
Limitation	n/a	

Scénario : Attendre la suppression d'un fichier

Ce scénario décrit un Job permettant de scanner un répertoire à la recherche d'un fichier qui sera supprimé de ce répertoire, pour ensuite lancer un sous-Job. Ainsi, lorsqu'un fichier est supprimé, le sous-Job affiche un message indiquant cette suppression.

- Ce Job est constitué de deux composants : le **tWaitForFile** et le **tMsgbox**.
- Cliquez et déposez ces composants dans l'espace de modélisation et connectez-les à l'aide d'un lien **Iterate** pour effectuer la boucle.
- Puis sélectionnez le composant **tWaitForFile**, et dans l'onglet **Basic settings** de la vue **Component**, paramétrez les propriétés de la condition et de la boucle :

- Dans le champ **Time (in seconds) between iterations**, paramétrez la durée en secondes entre chaque itération. Dans cet exemple, le répertoire sera scanné toutes les *5* secondes.
- Dans le champ **Max. number of iterations (infinite loop if empty)**, saisissez le nombre d'itération à effectuer avant que le Job ne soit arrêté. Dans cet exemple, le répertoire est scanné *5* fois maximum.
- Dans le champ **Directory to scan**, renseignez le chemin d'accès au répertoire devant être scanné.
- Dans le champ **Trigger action when**, sélectionnez la condition à rencontrer pour que le sous-Job soit lancé. Dans cet exemple, la condition est la suppression (ou le déplacement) d'un fichier du répertoire.
- Dans le champ **Then**, sélectionnez l'action à effectuer lorsque la condition est rencontrée avant que le nombre d'itérations défini ne soit atteint. Dans cet exemple, la boucle s'arrête dès que la condition est rencontrée.
- Puis définissez le sous-Job à effectuer lorsque la condition est rencontrée. Dans cet exemple, le sous-Job affiche une boîte de dialogue.
- Sélectionnez le composant **tMsgBox**, et dans l'onglet **Basic settings** de la vue **Component**, saisissez le message à afficher.
- Dans les champs **Title** et **Message**, renseignez le titre et le message.
- Dans les champs **Buttons** et **Icon**, renseignez le type de boutons et d'icônes à utiliser.

- Dans le champ **Message**, vous pouvez saisir n'importe quel type de message que vous souhaitez faire apparaître et utiliser des variables globales disponibles dans la liste d'auto-complétion via le raccourci **Ctrl+Espace**.
- Par exemple, en Perl, le message utilisé pour ce Job est : `"Deleted File: $_globals{tWaitForFile_1}{DELETED_FILE}, on Iteration : $_globals{tWaitForFile_1}{CURRENT_ITERATION}\n"`
- Le message en Java est : `"Deleted file: "+((String)globalMap.get("tWaitForFile_1_DELETED_FILE"))+" on iteration Nr:"+((Integer)globalMap.get("tWaitForFile_1_CURRENT_ITERATION"))`

Exécutez le Job via la raccourci **F6**. Pendant que la boucle s'effectue, supprimez un fichier du répertoire défini et la boîte de dialogue contenant le message défini apparaît.

tWaitForSocket

Propriétés du tWaitForSocket

Famille de composant	Orchestration	
Fonction	Le composant **tWaitForSocket** fait une boucle sur un port défini à la recherche de données puis déclenche un sous-job lorsque la condition est rencontrée.	
Objectif	Ce composant permet de déclencher un Job en fonction d'une condition définie.	
Basic settings	*Port*	Saisissez le numéro du port à écouter.
	End of line separator	Saisissez le séparateur de fin de ligne de vos données.
	Then	Sélectionnez l'action à effectuer : soit continuer à écouter le port (**keep on listening**) soit fermer la connexion (**close socket**).
	Print client/server data	Cochez cette case afin d'afficher les données client ou serveur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est un composant de début (ou de déclenchement) pour le sous-Job exécuté en fonction de la condition établie. Ainsi, ce composant doit être connecté à un sous-job via un lien Iterate.	
Global Variables		**Client input data** : retourne les données transmises par le client. Disponible comme variable **Flow**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Limitation	n/a	

Scénario associé

Aucun scénario n’est disponible pour ce composant.

tWaitForSqlData

Propriété du tWaitForSqlData

Famille de composant	Orchestration	
Fonction	Le composant **tWaitForSqlData** effectue une boucle sur une connexion donnée à la recherche d'un ajout ou d'une suppression de lignes, puis déclenche un sous-Job lorsque la condition est rencontrée.	
Objectif	Ce composant permet de déclencher un sous-Job en fonction de la présence de données sql.	
Basic settings	*Wait at each iteration (in seconds)*	Paramétrez en secondes l'intervalle de temps entre chaque vérification.
	Max. iterations (infinite loop if empty)	Nombre de vérifications à effectuer avant que le Job ne s'arrête (si vous ne saisissez rien, la boucle sera infinie).
	Use an existing connection/Component List	Cochez la case et cliquez sur le composant de connexion adéquat dans la liste **Component list** pour réutiliser les informations de connexion que vous avez déjà définies. Lorsqu'un Job contient un Job parent et un Job enfant, la liste **Component List** présente uniquement les composants de connexion du Job du même niveau. Si vous souhaitez utiliser une connexion existant dans un autre niveau, vérifiez que les composants de connexion disponibles partagent la même connexion. Pour plus d'informations concernant le partage d'une connexion à travers différents niveaux de Jobs, consultez **Use or register a shared DB connection** dans tous les composants de base de données que vous utilisez. Sinon, vous pouvez également désactiver le composant de connexion et utiliser les **Dynamic settings** du composant, afinde paramétrer manuellement la connexion. Dans ce cas, vérifiez que le nom de la connexion est unique et distinct tout au long des deux niveaux de Jobs. Pour plus d'informations concernant les paramètres dynamiques (**Dynamic settings**), consultez le **Guide Utilisateur** de votre Studio.
	Table to scan	Nom de la table à vérifier.

	Trigger action when rowcount is	Sélectionnez la condition à rencontrer pour exécuter l'action : **Equal to** : égal à **Not Equal to** : différent de **Greater than** : supérieur à **Lower than** : inférieur à **Greater or equal to** : supérieur ou égal à **Lower or equal to** :inférieur ou égal à.
	Value	Définissez la valeur à prendre en compte.
	Then	Sélectionnez l'action à effectuer lorsque la condition est rencontrée : soit arrêter la boucle (**exit loop**), soit continuer la boucle jusqu'à ce que le nombre d'itération maximum soit atteint (**continue loop**).
Utilisation	Bien que ce composant ait besoin d'un composant de connexion pour ouvrir l'accès à une base de données, il joue aussi le rôle de composant de début (ou de déclenchement) pour le sous-Job à exécuter lorsque la condition est rencontrée. Ainsi, ce composant doit être connecté à un sous-Job via un lien Iterate.	
Global Variables		**Current iteration** : indique le nombre d'itérations. Disponible comme variable **Flow**. Retourne un entier. **Row count** : indique le nombre d'enregistrements détectés dans la table. Disponible comme variable **Flow**. Retourne un entier. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur Talend Open Studio**.
Limitation	n/a	

Scénario : Attendre l'ajout de lignes dans une table

Ce scénario décrit un Job lisant la table d'une base de données dans l'attente d'un ajout de données dans cette table, pour ensuite exécuter un sous-Job. Lorsque des données sont ajoutées, le sous-Job effectue un Select* sur la table et affiche le contenu des données insérées dans la console standard. Cet exemple est présenté en Perl, mais les paramètres sont les mêmes si vous souhaitez l'effectuer en Java.

- Cliquez et déposez les composants suivants de la **Palette** dans l'espace de modélisation : **tMysqlConnection**, **tWaitForSqlData**, **tMysqlInput**, **tLogRow**.

- Connectez le composant **tMysqlConnection** au composant **tWaitforSqlData** à l'aide un lien **OnSubJobOK**, disponible via le clic-droit.
- Puis connectez le composant **tWaitForSqlData** au sous-Job à l'aide d'un lien **Iterate** puisqu'aucune donnée n'est transférée pour le moment. En effet, le **tWaitForSqlData** effectue seulement des boucles jusqu'à ce que la condition soit rencontrée.
- Dans le sous-Job à exécuter si la condition est rencontrée, le composant **tMysqlInput** est connecté au **tLogRow**. Puisque cette connexion sert à transférer des données, utilisez un lien de type **Row main**.
- Maintenant, paramétrez la connexion à la table devant être vérifée à intervalles réguliers. Dans l'onglet **Basic Settings** de la vue **Component** du composant **tMySqlConnection**, paramétrez les propriétés de connexion à la base de données.

- Renseignez les champs **Host**, **Port**, **Database**, **Username**, **Password** pour ouvrir une connexion à la base de données. Pour plus d'informations, consultez les composants de connexion aux bases de données. Par exemple, pour une base de données Mysql, consultez *tMysqlConnection, page 471*.
- Dans le champ **Encoding**, sélectionnez l'encodage adéquat, si nécessaire.
- Puis sélectionnez le composant **tWaitForSqlData**, et dans l'onglet **Basic Settings** de la vue **Component**, paramétrez ses propriétés.
- Dans le champ **Wait at each iteration**, paramétrez en secondes l'intervalle entre chaque itération.

- Dans le champ **Max iterations**, saisissez le nombre d'itérations maximum à effectuer avant que le Job ne s'arrête.
- Pour utiliser le composant **tWaitForSqlData**, il faut qu'une connexion soit ouverte pour effectuer le nombre de boucles défini. Sélectionnez la connexion adéquate (s'il y en a plusieurs) dans la liste déroulante **Component List**.

- Dans le champ **Table to scan**, saisissez le nom de la table à scanner. Dans cet exemple : *test_datatypes*.
- Dans les champs **Trigger action when rowcount is** et **Value**, sélectionnez la condition à rencontrer afin de lancer le sous-Job. Pour cet exemple, le nombre de lignes à prendre en compte dans la table scannée soit être supérieur ou égal à 1, ainsi sélectionnez *be greater or equal to 1*.
- Dans le champ **Then**, sélectionnez l'action à effectuer lorsque la condition est rencontrée avant que le nombre maximum d'itérations défini ne soit atteint. Dans cet exemple, dès que la condition est rencontrée, la boucle devrait s'arrêter.
- Puis paramétrez le sous-Job à exécuter lorsque la condition est rencontrée. Dans cet exemple, le sous-Job sélectionne les données de la table scannée et les affiche dans le console.
- Sélectionnez le composant **tMysqlInput**, et dans l'onglet **Basic Settings** de la vue **Component**, paramétrez la connexion à la table.

- Si la connexion est stockée dans le Repository, sélectionnez-la dans la liste déroulante. Sinon, cochez la case **Use an existing connection** et sélectionnez le composant de connexion correspondant dans la liste.
- Dans cet exemple, le schéma correspondant à la structure de la table est stocké dans le **Repository**.
- Dans le champ **Table Name**, renseignez le nom de la table à partir de laquelle les données seront extraites, *Test_datatypes*.
- Puis dans le champ **Query**, saisissez la commande Select permettant d'extraire le contenu de la table.
- Pour cet exemple, laissez les propriétés par défaut du composant **tLogRow**.

Avant d'exécuter le Job, assurez-vous que la table à scanner (*test_datatypes*) est bien vide, afin que la condition (*greater or equal to 1*) soit rencontrée. Puis exécutez le Job via le raccourci **F6**. Avant la fin de la boucle, ajoutez une ou deux lignes à la table test_datatypes afin que la condition soit rencontrée.

id	log_msg
2	143.112.32.4 - - [04/Mar/2008:00:00:00 +0100] "GET /c...
1	143.112.32.4 - - [04/Mar/2008:00:00:00 +0100] "GET /c...

Le Job s'arrête lorsque les données ajoutées à la table sont détectées lors de la boucle et le contenu de la table est affiché dans la console.

```
Starting job tWaitForSql at 16:55 06/03/2008.
2| 143.112.32.4 - - [04/Mar/2008:00:00:00 +0100] "GET
/components/none HTTP/1.1" 404 354 "-" "Mozilla/4.0 (compatible
1| 143.112.32.4 - - [04/Mar/2008:00:00:00 +0100] "GET
/components/none HTTP/1.1" 404 354 "-" "Mozilla/4.0 (compatible
Job tWaitForSql ended at 16:55 06/03/2008. [exit code=0]
```

Composants Processing

Ce chapitre passe en revue les propriétés des principaux composants présents dans la famille **Processing** de la **Palette** de **Talend Open Studio**.

La famille **Processing** regroupe les composants vous permettant d'effectuer différents types de traitement de flux de données, notamment l'agrégation, le mapping, la transformation, la dénormalisation, le filtrage, etc.

tAggregateRow

Propriétés du tAggregateRow

Famille de composant	Processing	
Fonction	Le composant **tAggregateRow** reçoit un flux de données et fait une agrégation basée sur une ou plusieurs colonnes. Pour chacune des lignes en sortie, une clé d'agrégation est fournie, ainsi que le résultat de l'opération d'agrégation correspondant (min, max, sum, etc.).	
Objectif	Il permet d'établir des métriques et des statistiques basées sur des valeurs ou des calculs.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que lorsque vous modifiez le schéma, il se met automatiquement en built-in.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Group by	Définit les ensembles d'agrégation, dont les valeurs sont utilisées pour les calculs.
		Output Column : Sélectionnez le libellé de colonne dans la liste fournie, basée sur la structure de schéma que vous avez définie. Vous pouvez ajouter autant de colonnes de sortie que vous le souhaitez afin d'affiner les agrégations. Ex : Sélectionnez *Country* (Pays) pour calculer la moyenne des valeurs pour chaque pays ou sélectionnez *Country* et *Region* pour comparer les résultats des régions d'un pays par rapport aux régions d'un autre pays.
		Input Column : Faites la correspondance entre les libellés des colonnes d'entrée avec ceux des colonnes de sortie, dans le cas où vous souhaitez que les libellés du schéma de sortie soient différents du schéma d'entrée.
	Operations	Sélectionnez le type d'opération à effectuer ainsi que la valeur à utiliser pour le calcul et le champ de sortie.

		Output Column : Sélectionnez le champ de destination dans la liste.
		Fonction : Sélectionnez l'opérateur parmi : **count :** calcule le nombre de ligne, **min :** sélectionne la plus petite valeur, **max :** sélectionne la plus grande valeur, **avg :** calcule la moyenne, **sum :** calcule la somme, **first :** retourne la première valeur, **last :** retourne la dernière valeur, **list :** liste les valeurs d'une agrégation de plusieurs clés. **list (object) :** liste les valeurs Java d'une agrégation de plusieurs clés, **count (distinct) :** compte le nombre de lignes sans les doublons, **standard deviation :** calcule l'écart-type.
		Input column position : Sélectionnez la colonne d'entrée à partir de laquelle les valeurs sont collectées pour l'agrégation.
		Ignore null values : Cochez cette case devant le nom de toutes les colonnes pour lesquelles vous voulez que les valeurs nulles soient ignorées.
Advanced settings	*Delimiter(only for list operation)*	Saisissez le séparateur que vous souhaitez utiliser afin de séparer les différentes opérations.
	Use financial precision, this is the max precision for "sum" and "avg" operations, checked option heaps more memory and slower than unchecked.	Cochez cette case afin d'utiliser une précision financière, une précision maximale, mais qui consomme plus de mémoire et rend le processus plus lent. *Nous vous conseillons d'utiliser le type BigDecimal en sortie avec cette option, afin d'obtenir des résultats précis.*
	Check type overflow (slower)	Vérifie les types des données afin d'éviter que le Job ne plante
	Check ULP (Unit in the Last Place), ensure that a value will be incremented or decremented correctly, only float and double types. (slower)	Cochez cette case afin d'avoir la meilleure précision possible pour les types Float et Double.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est un composant intermédiaire car il traite un flux de données, par conséquent il requiert des composants en entrée comme en sortie. Généralement, l'utilisation du composant **tAggregateRow** est combinée au composant **tSortRow**.	
Limitation	n/a	

Scénario : Agréger des valeurs et trier des données

Le scénario suivant décrit un Job à quatre composants. Le composant d'entrée, un fichier délimité csv, contient des noms de pays et des valeurs de notation à trier par ordre décroissant de moyenne. Ce composant d'entrée est connecté à un opérateur **tAggregateRow** qui se charge de calculer les moyennes puis à un composant **tSortRow** qui se charge du tri. Le flux de sortie est dirigé dans un nouveau fichier CSV.

- A partir du dossier **File** de la Palette, cliquez et déposez un composant **tFileInputDelimited**.
- Cliquez sur le libellé du composant et renommez-le en *Countries*. Ou passez dans l'onglet **View** pour le changer.
- Dans l'onglet **Basic settings**, définissez le chemin d'accès ainsi que les critères de délimitation. Ou sélectionnez l'entrée de métadonnées correspondant à votre fichier d'entrée, si vous en avez défini une.
- Cliquez sur **Edit schema...** et paramétrez les colonnes : *Countries* et *Points* correspondant à la structure du fichier. Si la description du fichier est conservée dans les **Metadata** du Repository, le schéma est automatiquement chargé lorsque vous cliquez sur **Repository** dans le champ **Schema type**.
- Puis dans le dossier **Processing** de la **Palette**, cliquez et déposez le composant **tAggregateRow**. Renommez-le en *Calculation*.
- Cliquez-droit et connectez *Countries* à *Calculation* à l'aide d'une connexion **Row > Main**.
- Double-cliquez sur *Calculation* (composant **tAggregateRow**) pour définir les propriétés. Cliquez sur **Edit schema** et définissez le schéma de sortie. Vous pouvez ajouter autant de colonnes que vous le voulez selon le nombre d'opérations d'ensemble que vous souhaitez obtenir en sortie.

- Dans cet exemple, calculez la moyenne de la notation par pays et affichez les valeurs minimale et maximale par pays, sachant que chaque pays possède plusieurs notes. Cliquez sur **OK** lorsque le schéma est terminé.
- Définissez à présent les différentes opérations d'ensemble à effectuer. Dans le champ **Group By** de la fenêtre **Basic settings** du composant **tAggregateRow**, définissez les ensembles sur lesquels les opérations sont effectuées. Dans cet exemple, sélectionnez **Country** comme colonne de regroupement. Notez que dans la colonne de sortie, un champ clé doit être défini dans le schéma. La première colonne citée comme de sortie (Output) dans la table **Group By** devient l'ensemble principal de calcul. Toutes les autres sorties deviennent alors secondaires dans l'ordre d'affichage.
- Sélectionnez la colonne d'entrée (Input) dont sont extraites les valeurs.
- Puis renseignez les diverses opérations réalisées. Les fonctions utilisées dans cet exemple sont des moyennes (average), min, max. Sélectionnez la colonne d'entrée dont sont extraites les valeurs de calcul et sélectionnez les cases **Ignore null values** si nécessaire.

- Cliquez-déposez un composant **tSortRow** dans l'espace de modélisation. Pour plus d'informations concernant ce composant, consultez *Propriétés du tSortRow, page 1346.*
- Connectez-le au composant **tAggregateRow** à l'aide d'une connexion de type **Row > Main**.
- Dans la vue **Component** du composant **tSortRow**, définissez la colonne de tri, le type ainsi que l'ordre de tri.

- Dans ce scénario, la colonne à trier est *Country*, le type de tri est alphabétique et l'ordre est ascendant.
- Ajoutez un composant de sortie à ce Job. Cliquez et déposez un composant **tFileOutputDelimited** et paramétrez-le.
- Connectez le composant **tSortRow** à ce composant de sortie.

- Dans la vue **Component**, saisissez le chemin d'accès au fichier de sortie. Cliquez sur **Edit schema** pour le configurer si nécessaire. Dans ce scénario, le fichier délimité est de type CSV. Cochez la case **Include Header** afin de réutiliser les libellés des colonnes du schéma dans le flux de sortie.
- Appuyez sur **F6** pour exécuter le Job. Le fichier CSV est ainsi créé et il contient les résultats des opérations d'agrégation.

tAggregateSortedRow

Propriétés du tAggregateSortedRow

Famille du composant	Processing	
Fonction	Le composant **tAggregateSortedRow** reçoit un flux de données triées sur lequel il effectue une agrégation basée sur une ou plusieurs colonnes. La clé d'agrégation et les résultats pertinents des opérations (min, max, sum, etc.) sont dispensés pour chaque ligne de sortie.	
Objectif	Le **tAggregateSortedRow** vous permet de récupérer un échantillon de métriques reposant sur des valeurs ou des calculs. Etant donné que les données du flux d'entrée sont supposées être déjà triées, les performances n'en sont que plus optimisées.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que lorsque vous modifiez le schéma, il se met automatiquement en built-in.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Group by	Définit les ensembles d'agrégation, dont les valeurs sont utilisées pour les calculs.
		Output Column : Sélectionnez le libellé de colonne dans la liste fournie, basée sur la structure de schéma que vous avez définie. Vous pouvez ajouter autant de colonnes de sortie que vous le souhaitez afin d'affiner les agrégations. Ex : Sélectionnez *Country* (Pays) pour calculer la moyenne des valeurs pour chaque pays ou sélectionnez *Country* et *Region* pour comparer les résultats des régions d'un pays par rapport aux régions d'un autre pays.
		Input Column : Faites la correspondance entre les libellés des colonnes d'entrée avec ceux des colonnes de sortie, dans le cas où vous souhaitez que les libellés du schéma de sortie soient différents du schéma d'entrée.

	Operations	Sélectionnez le type d'opération à effectuer ainsi que la valeur à utiliser pour le calcul et le champ de sortie.
		Output Column : Sélectionnez le champ de destination à partir de la liste déroulante.
		Function : Choisissez l'opérateur parmi : count, min, max, avg, first, last.
		Input column position : Sélectionnez la colonne d'entrée à partir de laquelle les valeurs sont récupérées avant d'être agrégées.
		Ignore null values : Cochez cette case devant le nom de toutes les colonnes pour lesquelles vous voulez que les valeurs nulles soient ignorées.
Utilisation	Ce composant traite des flux de données, il nécessite donc des composants d'entrée et sortie. Il est donc défini en tant que composant intermédiaire.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Agréger des valeurs et trier des données, page 1250* du composant **tAggregateRow**.

tConvertType

Propriétés du tConvertType

Famille de composant	Processing	
Fonction	Le composant **tConvertType** permet d'effectuer des conversions spécifiques du type de données Java vers un autre type de données Talend au cours de l'exécution.	
Objectif	Le composant **tConvertType** permet de convertir automatiquement des données de type Java vers un autre type utilisé par **Talend**, ce qui permet d'éviter les erreurs de compilation.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que lorsque vous modifiez le schéma, il se met automatiquement en built-in.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs designs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Auto Cast	Cette case est cochée par défaut. Elle permet de convertir automatiquement des données de type Java.
	Manual Cast	Ce mode n'est pas visible si la case **Auto Cast** est cochée. Elle permet de préciser manuellement les colonnes où une conversion de type Java est nécessaire.
	Set empty values to Null before converting	Cochez cette case pour définir les valeurs des éléments de type String ou Object comme null pour les données d'entrée.
	Die on error	Cette case est cochée pour arrêter le Job lorsque survient une erreur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant ne peut être utilisé comme composant de début. Il requiert un flux d'entrée pour fonctionner.	
Limitation	n/a	

Scénario : Convertir des types de données Java

Le scénario Java suivant permet de créer un Job à quatre composants dans lequel le **tConvertType** est utilisé pour convertir des données de type Java stockées dans trois colonnes différentes et le **tMap** est utilisé pour modifier le schéma : la première colonne reste intacte et les deux dernières colonnes du schéma d'entrée sont fusionnées en une seule colonne dans le schéma de sortie.

Dans ce scénario, le schéma du fichier délimité utilisé en entrée est stocké dans le Repository, vous pouvez donc renseigner les propriétés du composant **tFileInputDelimited** d'un simple glisser-déposer du répertoire **Repository** > **Metadata** > **File delimited** vers l'espace de modélisation. Pour plus d'informations, consultez *Déposer des composants à partir du répertoire Metadata* du **Guide utilisateur** de **Talend Open Studio**.

- Cliquez et déposez les composants suivants dans l'espace de modélisation : **tConvertType**, **tMap** et **tLogRow**.
- Dans le Repository, développez les noeuds **Metadata > File Delimited** et glissez-déposez la métadonnée correspondante dans l'espace de modélisation. Pour ce scénario, *JavaTypes*. La boîte de dialogue **[Components]** apparaît.
- Dans la boîte de dialogue, sélectionnez le **tFileInputDelimited** dans la liste des composants et cliquez sur **Ok**.
  Un composant **tFileInputComponent** appelé *Java types* apparaît dans l'espace de modélisation.
- Connectez les composants à l'aide de liens **Row Main**.

- Dans l'espace de modélisation, sélectionnez le composant **tFileInputDelimited** et cliquez sur la vue **Component** pour paramétrer ses propriétés de base.
- Dans l'onglet **Basic settings**, sélectionnez l'option **Repository** dans la liste **Property Type** puisque les propriétés du fichier sont stockées dans le **Repository**. Les autres champs sont alors renseignés automatiquement grâce aux données récupérées.

Le fichier d'entrée utilisé pour ce scénario est un fichier texte appelé *input* contenant des données Java de type string (chaîne de caractères), integer (entier) et float.

- Dans la vue **Basic settings**, renseignez les autres champs, si nécessaire. Pour plus d'informations, consultez les *Propriétés du tFileInputDelimited, page 915*. Dans ce scénario, il n'y a pas d'en-tête et de pied de page et il n'y a pas de nombre limite de lignes à traiter.
- Cliquez sur **Edit schema** pour décrire la structure des données du fichier d'entrée. Dans ce scénario, le schéma est composé de trois colonnes : *StringtoInteger, IntegerField* et *FloatToInteger*.

- Cliquez sur **OK** pour fermer la boîte de dialogue.
- Dans l'espace de modélisation, sélectionnez le **tConvertType** et cliquez sur la vue **Component** pour paramétrer ses propriétés de base.

- Dans la liste **Schema Type**, sélectionnez l'option **Built in** et cliquez sur le bouton **Sync columns** pour récupérer automatiquement les colonnes du composant **tFileInputDelimited**.
- Si nécessaire, cliquez sur **Edit schema** pour décrire manuellement la structure des données de ce composant de traitement.

Dans ce scénario, l'objectif est de convertir les données de type string (chaîne de caractères) en integer (entier) et des données de type float en integer.

- Cliquez sur **OK** pour fermer la boîte de dialogue **[Schema of tConvertType]**.
- Dans l'espace de modélisation, double-cliquez sur le **tMap** pour ouvrir l'éditeur du **tMap**. L'éditeur du **tMap** s'ouvre et affiche les noms des colonnes du fichier d'entrée.

- Dans l'onglet **Schema editor** en bas de l'éditeur du **tMap**, cliquez sur le bouton **[+]** de la table de sortie pour ajouter deux lignes et nommez-les *StringToInteger* et *Sum*.
- Dans l'éditeur du **tMap**, glissez la ligne *StringToInteger* de la table d'entrée vers la ligne *StringToInteger de la table de sortie*.
- Dans l'éditeur du **tMap**, glissez les lignes *IntegerField* et *FloatToInteger* de la table d'entrée vers la ligne *Sum* de la table de sortie.
- Cliquez sur **OK** pour fermer l'éditeur du tMap.

- Dans l'espace de modélisation, sélectionnez le **tLogRow** et cliquez sur la vue **Component** pour paramétrer ses propriétés de base. Pour plus d'informations, consultez le *tLogRow, page 1163*.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job tConvertType at 18:03 19/11/2008.
.---------------+---.
|      tLogRow_1      |
|=---------------+--=|
|StringToInteger|sum|
|=---------------+--=|
|3              |579|
'---------------+---'

Job tConvertType ended at 18:03 19/11/2008. [ex
```

Les données de type string sont converties en integer et affichées dans la colonne *StringToInteger* dans la console. Les données de type float sont converties en integer et ajoutées à la valeur *IntegerField* et leur somme est affichée dans la colonne *Sum* dans la console.

tDenormalize

Propriétés du tDenormalize

Famille de composant	Processing/Fields	
Fonction	Le composant **tDenormalize** dénormalise un flux entrant en fonction d'une colonne.	
Objectif	Le **tDenormalize** permet de synthétiser un flux entrant.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (**built-in**) soit distant dans le **Repository**. En Perl, le schéma est en lecture seule et obligatoirement en **Built-In**.
		Built-in : Le schéma est créé et conservé pour ce composant uniquement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
Java uniquement	*To denormalize*	Ce tableau vous permet de choisir les paramètres de dénormalisation des colonnes. **Column** : Sélectionnez la colonne à dénormaliser. **Delimiter** : Saisissez, entre guillemets doubles, le séparateur que vous souhaitez utiliser. **Merge same value** : Cochez cette case si vous voulez fusionner les valeurs identiques.
Perl uniquement	*Column to denormalize*	Sélectionnez la colonne du flux entrant sur laquelle est basée la dénormalisation (inclus dans la clé).
Java uniquement	*Group by*	Sélectionnez une ou plusieurs colonnes à regrouper. Nous vous conseillons de supprimer du schéma les colonnes qui ne sont pas utilisées avant de procéder au traitement.
Perl uniquement	*Item Separator*	Renseignez le séparateur délimitant les données du flux dénormalisé.
Advanced settings		
Perl uniquement	*Deduplicate items*	Supprime les doublons lorsque les valeurs dénormalisées sont concaténées.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en étape intermédiaire dans un flux de données.	
Limitation	Notez que ce composant peut changer l'ordre du flux entrant en Java et en Perl.	

Scénario 1 : Dénormaliser une colonne

Ce scénario illustre un Job dénormalisant une colonne d'un fichier délimité via un composant de dénormalisation, et affichant le résultat dans la console.

- Cliquez et déposez les composants suivants : **tFileInputDelimited**, **tDenormalize**, **tLogRow**.
- Connectez ces composants à l'aide d'un lien **Row main**.
- Dans la vue **Component** du composant **tFileInputDelimited**, renseignez le chemin d'accès au fichier à dénormaliser.

- Dans les champs **Header**, **Row Separator** et **Field Separator**, renseignez respectivement le nombre de lignes d'en-tête, le type de séparateur de lignes et de champs.
- Le fichier d'entrée est composé de deux colonnes : *Fathers* et *Children*.

- Dans l'onglet **Basic settings** du **tDenormalize**, paramétrez la colonne contenant des valeurs multiples à regrouper. Dans le tableau **To denormalize**, sélectionnez la colonne *Children*.
- Dans cet exemple, la colonne *Children* est celle à dénormaliser.

- Dans le champ **Delimiter**, paramétrez le séparateur des valeurs regroupées. Attention, une seule colonne peut être dénormalisée.
- Cochez la case **Merge same value** si vous souhaitez supprimer les éventuels doublons.
- Enregistrez le Job et exécutez-le en appuyant sur **F6**.

```
Starting job tDenormalize at 10:50 09/03/2016.

Mickael|Océane
Stéphane|Agathe;Clémence
Pierrick|Erwann;Tiphaine
Robert|Manon
Richard|Roméo
Fabrice|Martin
Job tDenormalize ended at 10:50 09/03/2016. [exit code=0]
```

Toutes les valeurs de la colonne *Children* (définie comme colonne à dénormaliser) sont regroupées en fonction des valeurs de la colonne *Fathers*. Les valeurs sont séparées par des virgules.

Scénario 2 : Dénormaliser plusieurs colonnes

Ce scénario illustre un Job dénormalisant deux colonnes d'un fichier délimité via un composant de dénormalisation et affichant le résultat dans la console.

- Cliquez et déposez les composants suivants : **tFileInputDelimited**, **tDenormalize**, **tLogRow**.
- Connectez tous les composants à l'aide d'une connexion **Row main**.

- Dans l'onglet **Basic settings** du composant **tFileInputDelimited**, renseignez le chemin d'accès au fichier à dénormaliser.

- Renseignez les autres champs dont les champs **Row** et **Field separators** et **Header** contenant respectivement les séparateurs de lignes et de champs et le nombre de lignes d'en-tête.
- Le schéma du fichier est composé de quatre colonnes : **Name**, **FirstName**, **HomeCity**, **WorkCity**.

- Dans l'onglet **Basic settings** du composant **tDenormalize**, sélectionnez les colonnes contenant les répétitions. Elles correspondent aux colonnes apparaissant plusieurs fois dans le document. Dans cet exemple, *FirstName*, *HomeCity* et *WorkCity* sont les colonnes à partir desquelles la dénormalisation est effectuée.
- Ajoutez autant trois lignes à la table **To denormalize** que vous le souhaitez en cliquant sur le bouton **[+]**. Puis sélectionnez les colonnes *FirstName*, *HomeCity* et *WorkCity* dans la liste déroulante.

- Dans la colonne **Delimiter**, paramétrez les délimiteurs en les saisissant entre guillemets, afin de séparer les valeurs concaténées. Pour la colonne *FirstName*, saisissez “#”, pour la colonne *HomeCity*, saisissez “§”, et pour la colonne *WorkCity*, saisissez “¤”.
- Enregistrez le Job et exécutez-le, en appuyant sur **F6**.

```
Starting job tDenormalize2 at 12:03 09/03/2010.

Moore|Demi#Demi|New York§Rio de Janeiro|Paris¤Los Angeles
Joli|Angelina#Angelina#Angelina|Berlin§Berlin§Los
Angeles|Berlin¤Los Angeles¤Los Angeles
Pitt|Brad#Brad|Beverly Hills§Paris|Los Angeles¤London
Willis|Bruce#Bruce#Bruce#Bruce|Paris§Paris§Madrid§Roma|Los
Angeles¤Madrid¤Madrid¤Dublin
Job tDenormalize2 ended at 12:03 09/03/2010. [exit code=0]
```

- Le résultat affiche les valeurs dénormalisées concaténées à l’aide des séparateurs.
- Dans l’onglet **Basic settings** du composant **tDenormalize**, dans le tableau **To denormalize**, cochez la case **Merge same value** pour fusionner les valeurs identiques.
- Enregistrez le Job et exécutez-le en appuyant sur **F6**.

```
Starting job tDenormalize2 at 13:44 09/03/2010.

Moore|Demi|New York§Rio de Janeiro|Paris¤Los Angeles
Joli|Angelina|Berlin§Los Angeles|Berlin¤Los Angeles
Pitt|Brad|Beverly Hills§Paris|Los Angeles¤London
Willis|Bruce|Paris§Madrid§Roma|Los Angeles¤Madrid¤Dublin
Job tDenormalize2 ended at 13:44 09/03/2010. [exit code=0]
```

Cette fois-ci, la console affiche les résultats sans doublon.

tDenormalizeSortedRow

Propriétés du tDenormalizeSortedRow

Famille du composant	Processing/Fields	
Fonction	Le composant **tDenormalizeSortedRow** rassemble dans un groupe toutes les lignes d'entrée dénormalisées triées ainsi que leurs valeurs distinctes, associées par des séparateurs de champ.	
Objectif	Le **tDenormalizeSortedRow** permet de synthétiser les lignes d'entrée triées afin de faire des économies de mémoire.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Input rows count	Renseignez le nombre de lignes d'entrée.
	To denormalize	Renseignez le nom de la colonne à dénormaliser.
Utilisation	Ce composant traite des flux de données, il nécessite donc un composant d'entrée et un composant de sortie.	
Limitation	n/a	

Scénario : Regrouper des lignes triées

Ce scénario Java décrit un Job à quatre composants. Il permet de lire un fichier délimité donné ligne par ligne, de trier les données d'entrées en fonction de leur type et de leur ordre, de dénormaliser toutes les lignes d'entrées triées et enfin d'afficher le résultat dans la console de log **Run**.

- A partir de la **Palette**, cliquez-déposez les composants suivants dans l'éditeur graphique : le **tFileInputDelimited**, le **tSortRow**, le **tDenormalizeSortedRow** et le **tLogRow**.
- Connectez les quatre composants à l'aide de liens de type **Row Main**.

- Dans l'éditeur graphique, sélectionnez le composant **tFileInputDelimited**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tFileInputDelimited**.

- Définissez le champ **Property Type** en mode **Built-In**.
- Renseignez le chemin d'accès au fichier à traiter dans le champ **File Name**. Dans cet exemple, on utilise le fichier *name_list*, qui comporte deux colonnes, *id* et *first name*.

- Définissez si nécessaire les séparateurs de lignes et de champs, l'en-tête et le pied-de-page, ainsi que le nombre de lignes traitées.
- Définissez le champ **Schema** en mode **Built in** puis cliquez sur le bouton **[...]** à côté du champ **Edit Schema** pour définir les données à passer au composant suivant. Dans cet exemple, le schéma est constitué de deux colonnes, *id* et *name*.

- Dans l'espace graphique, sélectionnez le composant **tSortRow**.
- Cliquez sur la vue **Component** pour en définir la configuration de base (**Basic settings**) du **tSortRow**.

- Définissez le champ **Schema Type** en mode **Built-In** puis cliquez sur **Sync columns** pour récupérer le schéma à partir du composant **tFileInputDelimited**.
- Dans le panneau **Criteria**, cliquez sur le bouton **[+]** pour ajouter une ligne puis définissez les paramètres de tri pour la colonne du schéma à traiter. Dans cet exemple, on veut trier la colonne *id* dans l'ordre croissant.
- Dans l'espace graphique, sélectionnez le composant **tDenormalizeSortedRow**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tDenormalizeSortedRow**.

- Définissez le champ **Schema Type** en mode **Built-In** puis cliquez sur **Sync columns** pour récupérer le schéma à partir du schéma du composant **tSortRow**.
- Dans le champ **Input rows count**, entrez le nombre de lignes d'entrée à traiter ou cliquez simultanément sur **Ctrl+Espace** pour accéder à la liste des variables de contexte puis sélectionnez la variable : `tDenormalizeSortedRow_1.NB_LINE`.
- Dans le panneau **To denormalize**, cliquez sur le bouton **[+]** pour ajouter une ligne puis définissez les paramètres de la colonne à dénormaliser. Dans cet exemple, on veut dénormaliser la colonne *name*.
- Dans l'éditeur graphique, sélectionnez le composant **tLogRow** puis cliquez sur la vue **Component** pour en définir la configuration de base (**Basic settings**). Pour plus d'informations sur le **LogRow**, consultez le chapitre *tLogRow, page 1163*.
- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

```
Starting job denormalize1 at 14:42 01/09/2008.
bufferSize=100000 objects
Sorting buffer...
0 milliseconds for 9 objects to sort in memory. 2147483647
items/s
Writing ordered buffer in file...
16 milliseconds for 9 objects to write in file. 562  items/s
.--+-----------------------.
|        tLogRow_1          |
|=-+-----------------------=|
|id|name                    |
|=-+-----------------------=|
|1 | Harrison; Jerry        |
|2 | Ford                   |
|3 | James; Goldman         |
|4 | Bill; Ford; Harry; Sue|
'--+-----------------------'

Job denormalize1 ended at 14:42 01/09/2008. [exit code=0]
```

Le résultat affiché sur la console montre la façon dont la colonne *name* a été dénormalisée.

tEmptyToNull

Propriétés du tEmptyToNull

Famille du composant	Processing	
Fonction	Le composant **tEmptyToNull** transforme les champs vides d'un fichier ou d'une table en champs NULL dans une base de données.	
Objectif	Le **tEmptyToNull** permet de remplacer des champs vides par des champs non définis produisant des valeurs NULL dans le composant de sortie.	
Basic settings		Ce composant ne nécessite aucune configuration, il s'exécute automatiquement.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant intermédiaire dans un flux de données, il nécessite donc un composant en entrée et un composant en sortie. Le composant en sortie doit obligatoirement être de type base de données.	
Limitation	n/a	

Scénario : Remplacer des champs vides par des champs non-définis (NULL)

Ce scénario Perl décrit un Job à trois composants. Il permet de remplacer les champs sans chaînes de caractères par des champs non-définis afin de générer des valeurs NULL en sortie.

- A partir de la **Palette**, cliquez-déposez les composants suivants dans l'éditeur graphique : le **tFileInputDelimited**, le **tEmptyToNull** et le **tMysqlOutput**.
- Connectez les trois composants à l'aide de liens de type **Row Main**.

- Dans l'éditeur graphique, sélectionnez le composant **tFileInputDelimited**.
- Cliquez sur la vue **Component** pour configurer les paramètres de base (**Basic settings**) du **tFileInputDelimited**.

- Dans le champ **Property Type**, sélectionnez le mode **Repository** si vous avez stocké les informations du fichier délimité sous le noeud **Metadata** du **Repository** ou sélectionnez le mode **Built-In** pour les définir manuellement. Dans ce scénario, on utilise le mode **Built-In**.
- Renseignez le chemin d'accès au fichier à traiter dans le champ **File Name**. Dans cet exemple, on utilise le fichier *name_list*, qui comporte quatre colonnes, *id*, *first name*, *last name* et *login*.

- Définissez si nécessaire les séparateurs de lignes (**Row separator**) et de champs (**Field separator**), l'en-tête (**Header**) et le pied-de-page (**Footer**), ainsi que le nombre de lignes à traiter (**Limit**). La première ligne, qui porte le nom des colonnes, est ignorée dans cet exemple, c'est pourquoi le champ **Header** est défini à "1".
- Dans le champ **Schema**, sélectionnez le mode **Built in** puis cliquez sur le bouton **[...]** à côté du champ **Edit Schema** pour définir les données à passer au composant suivant. Dans cet exemple, le schéma est constitué de quatre colonnes, *id*, *first_name*, *last_name* et *login*.

- Dans l'éditeur graphique, double-cliquez sur le composant **tMysqlOutput** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component** :

- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent. Vous pouvez cliquer sur le bouton **[...]** à coté du champ **Edit schema** pour consulter le schéma.
- Dans le champ **Property Type**, sélectionnez **Repository** si vous avez stocké les informations de connexion à la base de données sous le noeud **Metadata** du **Repository** ou sélectionnez le mode **Built-In**, puis renseignez manuellement les champs suivants : **Host**, **Port**, **Database**, **Username** et **Password**. Pour plus d'informations sur les propriétés du composant **tMysqlOutput**, consultez *tMysqlOutput, page 489.*
- Dans le champ **Table**, saisissez le nom de la table dont les champs contiendront les données extraites du fichier délimité.
- Dans le champ **Action on table**, sélectionnez l'opération que vous souhaitez effectuer sur la table. Pour ce scénario, sélectionnez **Create table** pour créer la table.

- Dans le champ **Action on data**, sélectionnez l'opération que vous souhaitez effectuer sur les données. Pour ce scénario, laissez l'option **Insert**.
- Enregistrez le Job puis appuyez sur **F6** pour l'exécuter.

id	first_name	last_name	login
1	Georges	Harrison	gharrison
2	Paul	McCartney	pmccartney
3	NULL	Lennon	jlennon
4	Ringo	NULL	rstarr
5	Mick	Jagger	mjagger
6	Keith	Richards	NULL

A partir de votre explorateur de base de données, vous pouvez vérifier que la table *namelist* a bien été créée. Elle se présente sous la forme des quatre colonnes *id*, *first_name*, *last_name* et *login* du schéma défini en sortie et les champs vides du fichier de départ on été remplacés par la valeur NULL.

tExternalSortRow

Propriétés du tExternalSortRow

Famille du composant	Processing	
Fonction	Le composant **tExternalSortRow** utilise une application externe de tri pour trier les données d'entrée d'une ou plusieurs colonnes, en fonction de leur type et de leur ordre.	
Objectif	Le **tExternalSortRow** permet d'établir des métriques et des tables de classification.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	File Name	Nom du fichier à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide Utilisateur** de **Talend Open Studio**.
	Field separator	Caractère, chaîne ou expression régulière séparant les champs.
	External command "sort" path	Renseignez le chemin d'accès au fichier externe contenant l'algorithme de tri à utiliser.
	Criteria	Cliquez sur **[+]** pour ajouter autant de lignes que nécessaire pour mettre en place le tri. La première colonne définie dans le schéma est sélectionnée par défaut.
		Schema column : Sélectionnez la colonne de votre schéma sur laquelle vous souhaitez baser votre tri. Notez que l'ordre est important car il détermine la priorité de tri.
		Sort type : Numérique ou Alphabétique. D'autres types de tri sont à venir.

		Order : Ordre ascendant ou descendant.
Advanced settings	*Maximum memory*	Renseignez la taille de la mémoire physique que vous voulez allouer au processus de tri.
	Temporary directory	Définissez l'endroit où stocker les fichiers temporaires.
	Set temporary input file directory	Cochez la case pour afficher le champ dans lequel vous pouvez spécifier le répertoire du fichier d'entrée temporaire.
	Add a dummy EOF line	Cochez cette case lorsque vous utilisez le composant **tAggregateSortedRow**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant traite des flux de données, il nécessite donc un composant d'entrée et un composant de sortie.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le *Scénario : Trier des données, page 1347* du composant **tSortRow**.

tExtractDelimitedFields

Propriétés du tExtractDelimitedFields

Famille du composant	Processing/Fields	
Fonction	Le composant **tExtractDelimitedFields** génère des colonnes multiples à partir d'une colonne donnée dans un fichier délimité.	
Objectif	Le **tExtractDelimitedFields** permet d'extraire les "champs" à partir d'une chaîne de caractères, pour les écrire à un autre endroit par exemple.	
Basic settings	*Field to split*	Sélectionnez à partir de la liste le champ à séparer.
	Field separator	Définissez le séparateur de champs. La syntaxe regex utilise un certain nombre de caractères spéciaux comme opérateurs. Si vous utilisez l'un de ces opérateurs comme séparateur de champs, il est nécessaire de protéger le caractère par un double backslash.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Advanced separator(for number)*	Cochez cette option pour modifier les séparateurs utilisés pour les nombres.

	Trim columns	Cochez cette case pour supprimer les espaces en début et en fin de champ dans les colonnes sélectionnées.
	Check each row structure against schema	Cochez cette case pour synchroniser chaque ligne avec le schéma d'entrée.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du Job, ainsi qu'au niveau de chaque composant.
Utilisation	Ce composant traite des flux de données, il nécessite donc un composant d'entrée et un composant de sortie. Il permet d'extraire des données d'un champ délimité à l'aide d'un lien **Row** > **Main**, et permet de créer un flux de rejet avec un lien **Row** > **Reject**, filtrant les données dont le type ne correspond pas au type défini.	
Limitation	n/a	

Scénario : Extraire des champs à partir d'un fichier délimité par des virgules

Ce scénario Java décrit un Job à trois composants où le composant **tExtractdelimitedFields** est utilisé pour extraire deux colonnes à partir d'un fichier délimité par des virgules.

- A partir de la **Palette**, cliquez-déposez les composants suivants dans l'éditeur graphique : le **tFileInputDelimited**, le **tExtractDelimitedFields** et le **tLogRow**.
- Effectuez un clic-droit sur chacun des trois composants pour les connecter à l'aide de liens **Row Main**.

- Dans l'espace graphique, sélectionnez le composant **tFileInputDelimited**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tFileInputDelimited**.
- Dans l'onglet **Basic settings**, définissez le champ **Property Type** en mode **Built-In**.
- Cliquez sur le bouton **[...]** à côté du champ **File Name** pour définir le chemin d'accès au fichier d'entrée.

Le champ **File Name** est obligatoire.

Dans ce scénario, on utilise le fichier d'entrée *test5*. Il s'agit d'un fichier texte dont certaines données sont séparées par des virgules.

- Dans l'onglet **Basic settings**, renseignez les autres champs. Pour plus d'informations, consultez le chapitre *Propriétés du tFileInputDelimited, page 915*. Dans ce scénario, l'en-tête et le pied-de-page ne sont pas définis et aucune limite n'est renseignée concernant le nombre de lignes traitées.
- Cliquez sur **Edit schema** pour décrire la structure des données du fichier d'entrée. Dans ce scénario, le schéma est composé d'une seule colonne, *name*.

- Dans l'espace graphique, sélectionnez le composant **tExtractDelimitedFields**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tExtractDelimitedFields**.

- A partir de la liste déroulante **Field to split**, sélectionnez la colonne à dissocier. Dans ce scénario, il s'agit de la colonne *name*.
- Dans le champ **Field separator**, renseignez le séparateur de champs correspondant.
- Cliquez sur **Edit schema** pour décrire la structure des données de ce composant de traitement.
- Dans le panneau de sortie de la boîte de dialogue **[Schema of tExtractRegexFields]**, cliquez sur le bouton **[+]** pour ajouter deux colonnes, *firstname* et *lastname*, au schéma de sortie.

Dans ce scénario, on veut séparer la colonne *name* en deux colonnes dans le flux de sortie, *firstname* et *lastname*.

- Cliquez sur **OK** pour fermer la boîte de dialogue **[Schema of tExtractDelimitedFields]**.
- Dans l'espace graphique, sélectionnez le composant **tLogRow** puis cliquez sur la vue **Component** pour en définir la configuration de base (**Basic settings**). Pour plus d'informations, consultez le chapitre *tLogRow, page 1163*.
- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

```
Starting job extract_delimited at 16:39 25/08/2008.
.---------+--------.
|    tLogRow_1      |
|=--------+-------=|
|firstname|lastname|
|=--------+-------=|
|Janet    |Anderson|
|Martin   |Chairman|
|Lily     |Massy   |
'---------+--------'

Job extract_delimited ended at 16:39 25/08/2008. [exit
```

Les prénoms (*firstname*) et les noms (*lastname*) sont extraits et affichés sur la console dans leurs colonnes définies correspondantes.

tExtractPositionalFields

Propriétés du tExtractPositionalFields

Famille du composant	Processing/Fields	
Fonction	Le composant **tExtractPositionalFields** génère des colonnes multiples à partir d'une colonne donnée en utilisant des champs positionnels.	
Objectif	Le **tExtractPositionalFields** permet d'extraire les données d'un fichier positionnel selon un modèle (*pattern*).	
Basic settings	*Field*	Sélectionnez dans la liste le champ à partir duquel effectuer l'extraction.
	Customize	Cochez cette case pour personnaliser le format des données du fichier positionnel et renseignez les colonnes du tableau **Formats**. **Column** : Sélectionnez la colonne que vous souhaitez personnaliser. **Size** : Saisissez la taille correspondant à la colonne. **Padding char** : Saisissez entre guillemets le caractère de remplissage utilisé afin qu'il soit supprimé de votre champ. Le caractère par défaut est un espace. **Alignment** : Sélectionnez le paramètre d'alignement approprié.
	Pattern	Renseignez le modèle à utiliser pour l'extraction. Un modèle est constitué de valeurs de longueur séparées par des virgules et interprétées comme une chaîne de caractères entre guillemets. Assurez-vous que les valeurs saisies dans ces champs correspondent au schéma défini.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.

		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Advanced separator(for number)*	Cochez cette option pour modifier les séparateurs utilisés pour les nombres.
	Trim Column	Cochez cette case afin de supprimer les espaces en début et en fin de champ, dans toutes les colonnes.
	Check each row structure against schema	Cochez cette case pour synchroniser chaque ligne avec le schéma d'entrée.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du Job, ainsi qu'au niveau de chaque composant.
Utilisation	Ce composant traite des flux de données, il nécessite donc un composant d'entrée et un composant de sortie. Il permet d'extraire des données d'un champ délimité à l'aide d'un lien **Row** > **Main**, et permet de créer un flux de rejet avec un lien **Row** > **Reject**, filtrant les données dont le type ne correspond pas au type défini.	
Limitation	n/a	

Scénario associé

Pour un scénario associé, consultez le chapitre *tExtractRegexFields, page 1283*.

tExtractRegexFields

Propriétés du tExtractRegexFields

Famille du composant	Processing/Fields	
Fonction	Le composant **tExtractRegexFields** génère des colonnes multiples à partir d'une colonne donnée en utilisant des correspondances regex.	
Purpose	Le **tExtractRegexFields** permet d'utiliser des expressions régulières pour extraire les données selon un modèle.	
Basic settings	*Field to split*	Sélectionnez dans la liste le champ (ou colonne) à séparer.
	Regex	Renseignez l'expression régulière en fonction du langage de programmation que vous utilisez.
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Advanced settings	*Die on error*	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
	Check each row structure against schema	Cochez cette case pour synchroniser chaque ligne avec le schéma d'entrée.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du Job, ainsi qu'au niveau de chaque composant.

Utilisation	Ce composant traite des flux de données, il nécessite donc un composant d'entrée et un composant de sortie. Il permet d'extraire des données d'un champ délimité à l'aide d'un lien **Row** > **Main**, et permet de créer un flux de rejet avec un lien **Row** > **Reject**, filtrant les données dont le type ne correspond pas au type défini.
Limitation	n/a

Scénario : Extraire des noms, des domaines et domaines de premier niveau à partir d'adresses e-mail

Ce scénario Java décrit un Job à trois composants dans lequel le **tExtractRegexFields** est utilisé pour définir une expression régulière correspondant à l'une des colonnes des données d'entrée. Dans ce scénario, il s'agit de la colonne *email*. Le composant **tExtractRegexFields** est également utilisé pour trouver les correspondances avec ladite expression régulière. Cette expression comprend des identifieurs de champ permettant de séparer le nom d'utilisateur, le nom de domaine ainsi que le nom du domaine de premier niveau pour chaque adresse e-mail. Si l'adresse e-mail est valide, ces trois parties sont extraites et affichées sur la console dans trois colonnes différentes. Les données des deux autres colonnes d'entrée, *id* et *age*, sont également extraites puis redirigées.

- A partir de la **Palette**, cliquez-déposez les composants suivants dans l'espace graphique : le **tFileInputDelimited**, le **tExtractRegexFields** et le **tLogRow**.
- Connectez les trois composants à l'aide de liens **Row Main** comme ci-dessous.

- Dans l'espace graphique, cliquez sur le composant **tFileInputDelimited**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tFileInputDelimited**.
- A partir de l'onglet **Basic settings**, configurez le champ **Property Type** en mode **Built-In**.
- Cliquez sur le bouton [...] à côté du champ **File Name** pour définir le chemin d'accès au fichier d'entrée.

Le champ **File Name** est obligatoire.

Dans ce scénario, le fichier d'entrée utilisé est *test4*. Il s'agit d'un fichier texte comportant trois colonnes : *id*, *email* et *age*.

- Renseignez tous les autres champs comme demandé. Pour plus d'informations, consultez le chapitre *Propriétés du tFileInputDelimited, page 915*. Dans ce scénario, l'en-tête et le pied-de-page ne sont pas définis et aucune limite n'est renseignée concernant le nombre de lignes traitées.
- Cliquez sur le bouton **[...]** du champ **Edit schema** pour définir la structure des données du fichier d'entrée. Dans ce scénario, le schéma est composé de trois colonnes, *id*, *email* et *age*.
- Dans l'espace graphique, sélectionnez le composant **tExtractRegexFields**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tExtractRegexFields**.
- A partir de la liste déroulante **Field to split**, sélectionnez la colonne à séparer, à savoir *email* dans ce scénario.
- Dans le panneau **Regex**, renseignez l'expression régulière à utiliser pour effectuer la correspondance. Dans ce scénario, il s'agit d'une expression régulière Java.

- Cliquez sur **Edit schema** pour décrire la structure des données du composant de traitement.
- Dans le panneau **Output** de la boîte de dialogue **[Schema of tExtractRegexFields]**, cliquez sur le bouton **[+]** pour ajouter cinq colonnes au schéma de sortie.

Dans ce scénario, on veut séparer la colonne d'entrée *email* en trois colonnes *name*, *domain* et *tld* dans le flux de sortie. Les deux autres colonnes d'entrée seront extraites telles quelles.

- Cliquez sur **OK** pour fermer la boîte de dialogue **[Schema of tExtractRegexFields]**.
- Dans l'espace graphique, sélectionnez le composant **tLogRow** puis cliquez sur la vue **Component** pour en définir la configuration de base (**Basic settings**). Pour plus d'informations, consultez le chapitre *tLogRow, page 1163*.
- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

```
Starting job extract_regex_mine at 10:46 29/08/2008.
.--+----+------+---+---.
|       tLogRow_1        |
|=-+----+------+---+--=|
|id|name|domain|tld|age|
|=-+----+------+---+--=|
|1 |name|domain|com|24 |
|2 |name|domain|net|31 |
|3 |name|domain|org|13 |
'--+----+------+---+---'

Job extract_regex_mine ended at 10:46 29/08/2008. [exit code=0]
```

Le composant **tExtractRegexFields** effectue une correspondance entre toutes les adresse e-mail données et l'expression régulière définie, puis extrait le nom d'utilisateur, le domaine et le domaine de premier niveau, qu'il affiche sur la console dans trois colonnes séparées. Les deux autres colonnes, *id* et *age*, sont extraites telles quelles.

tExtractXMLField

tExtractXMLField appartient à deux familles de composants : Processing et XML. Pour plus d'informations concernant le composant **tExtractXMLField**, consultez *tExtractXMLField, page 1454*.

tFilterColumns

Propriétés du tFilterColumns

Famille du composant	Processing	
Fonction	Le composant **tFilterColumns** opère des modifications spécifiques, établies à partir d'un mapping du nom des colonnes, sur un schéma défini.	
Objectif	Le **tFilterColumns** permet d'homogénéiser des schémas, soit en fonction de l'ordre des colonnes, soit en supprimant ou en ajoutant des colonnes.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
Utilisation	Ce composant n'est pas un composant de début (fond vert) et il nécessite un composant de sortie.	

Scénario associé

Pour plus d'informations concernant l'utilisation du composant **tFilterColumns**, consultez le *Scénario : Remplacement mutliple et filtrage de colonnes, page 1339* du composant **tReplace**.

tFilterRow

Propriétés du tFilterRow

Famille du composant	Processing	
Fonction	Le composant **tFilterRow** filtre des lignes d'entrée en définissant des conditions sur les colonnes sélectionnées.	
Objectif	Le composant **tFilterRow** aide à paramétrer les filtres sur les données source.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. *Le schéma est en lecture seule.*
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Logical operator used to combine conditions	Si vous souhaitez utiliser les modes de filtrage simple et avancé à la fois, sélectionnez l'opérateur permettant de combiner les deux modes.
	Conditions	Cliquez sur le bouton **[+]** pour ajouter autant de conditions que voulu. Les conditions sont exécutées les unes après les autres pour chaque ligne. **Input column** : Sélectionnez la colonne du schéma sur laquelle exécuter la fonction. **Function** : Sélectionnez une fonction dans la liste. **Operator** : Sélectionnez l'opérateur permettant de relier la colonne d'entrée et la valeur. **Value** : Renseignez la valeur filtrée, entre guillemets si besoin est.
	Use advanced mode	Cochez cette case lorsque l'opération à effectuer ne peut aboutir grâce aux fonctions standard proposées. Dans le champ texte, entrez l'expression régulière comme demandé.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.

Utilisation	Ce composant n'est pas un composant de début (fond vert) et il nécessite un composant de sortie.

Scénario : Filtrer une liste de noms et faire une recherche dans la liste

Le scénario suivant est un Job Java qui utilise une condition simple et une expression régulière pour filtrer une liste d'enregistrements. Le Job écrit deux tables de sortie : la première liste tous les enregistrements de langue italienne dont les prénoms contiennent moins de six caractères et la seconde liste tous les enregistrements rejetés. Un message d'erreur s'affichera dans la table pour chaque enregistrement rejeté, afin d'expliquer pourquoi l'enregistrement a été rejeté.

- A partir de la **Palette**, cliquez-déposez les composants suivants dans l'éditeur graphique : un **tFixedFlowInput**, un **tFilterRow** et un **tLogRow**.
- Connectez le **tFixedFlowInput** au **tFilterRow** à l'aide d'un lien **Row** > **Main**. Ensuite, connectez le **tFilterRow** au **tLogRow**, à l'aide d'un lien **Row** > **Filter**.
- Déposez de la **Palette** dans l'espace de modélisation graphique un nouveau **tLogRow** et renommez-le *reject*. Reliez le **tFilterRow** à ce **tLogRow** à l'aide d'un lien **Row** > **Reject**.
- Dans l'espace graphique, cliquez sur le composant **tFixedFlowInput** afin d'afficher sa vue **Basic settings** et définir ses propriétés.
- Sélectionnez l'option **Use Inline Content(delimited file)** dans la zone **Mode**.
- Cliquez sur le bouton **[...]** à côté du champ **File Name** pour définir le chemin d'accès au fichier d'entrée.

- Renseignez les séparateurs de lignes (**Row Separator**) et de champs (**Field Separator**). Le séparateur de lignes est constitué d'un retour chariot et le séparateur de champs d'un point-virgule.
- Dans la liste **Schema**, sélectionnez **Buil-In**. Le schéma n'est donc pas stocké dans le **Repository**.
- Cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour définir le schéma du fichier d'entrée. Dans cet exemple, le schéma se compose de quatre colonnes : *firstname*, *gender*, *language* et *frequency*. Dans la colonne **Type**, sélectionnez **String** pour les trois premières colonnes et **Integer** pour la colonne *frequency*.

- Cliquez sur **OK** pour valider et fermer l'éditeur. Une boîte de dialogue s'ouvre et vous demande si vous voulez propager les changements. Cliquez sur **Yes** (oui).
- Dans la zone de texte **Content**, saisissez votre texte en respectant votre schéma.
- Double-cliquez sur le composant **tFilterRow** pour afficher sa vue **Basic settings** et définir ses propriétés.

- Dans le tableau **Conditions**, renseignez les paramètres de filtre sur la colonne *firstname*.
- Dans la colonne **InputColumn**, sélectionnez *firstname*.
  Dans la colonne **Function**, sélectionnez **Length**.
  Dans la colonne **Operator**, sélectionnez **Lower than**.
- Dans la colonne **Value**, saisissez *6* afin de filtrer uniquement les noms dont la longueur est supérieure à six caractères.

  Dans la colonne **Value**, vous devez saisir vos valeurs entre guillemets doubles pour tous les types de données, sauf pour le type **Integer**, qui n'en nécessite pas.

- Pour implémenter la recherche sur les noms italiens, cochez la case **Use advanced mode** et saisissez l'expression régulière suivante, qui comprend le nom de la colonne dans laquelle chercher : `input_row.language.equals("italian")`.
- Pour combiner les deux conditions, simple et avancée, sélectionnez l'opérateur logique **And** dans la liste **Logical operator used to combine conditions**.
- Dans la vue **Basic settings** des composants **tLogRow**, sélectionnez l'option **Table (print values in cells of a table)** dans la zone **Mode**.
- Sauvegardez votre Job puis appuyez sur **F6** pour l'exécuter.

```
Starting job FilterRow at 18:32 16/12/2010.

[statistics] connecting to socket on port 3601
[statistics] connected
.---------+------+--------+---------.
|             tLogRow_2              |
|=--------+------+--------+--------=|
|firstname|gender|language|frequency|
|=--------+------+--------+--------=|
|romeo    |m     |italian |29       |
'---------+------+--------+---------'

.---------+------+--------------------+---------+-------------------------------------------------.
|                                                 reject                                            |
|=--------+------+--------------------+---------+-------------------------------------------------=|
|firstname|gender|language            |frequency|errorMessage                                     |
|=--------+------+--------------------+---------+-------------------------------------------------=|
|romain   |m     |french              |16       | firstname.length()<6 faild|advanced condition failed|
|roman    |m     |russian,polish,czech|55       |advanced condition failed                        |
|romano   |m     |italian41           |null     | firstname.length()<6 faild|advanced condition failed|
|romolo   |m     |italian             |0        | firstname.length()<6 faild                      |
|romulus  |m     |roman mythology     |42       | firstname.length()<6 faild|advanced condition failed|
'---------+------+--------------------+---------+-------------------------------------------------'
[statistics] disconnected
Job FilterRow ended at 18:32 16/12/2010. [exit code=0]
```

La première table liste les enregistrements contenant des prénoms italiens comprenant moins de six caractères et la seconde affiche les enregistrements ne correspondant pas à la condition de filtre. A chaque enregistrement rejeté correspond un message d'erreur expliquant la raison du rejet.

tMap

La fonction de schéma dynamique fonctionne uniquement dans **Talend Integration Suite Studio**.

Propriétés du tMap

Famille de composant	Processing	
Fonction	Le **tMap** est un composant avancé qui s'intègre à **Talend Open Studio** comme un plugin.	
Objectif	Le **tMap** transforme et dirige les données à partir d'une ou plusieurs source(s) et vers une ou plusieurs destinations.	
Basic settings	*Preview*	L'aperçu est un instantané des données du Mapper. Il n'est visible que lorsque les propriétés du Mapper sont renseignées. La synchronisation de l'aperçu ne prend effet qu'après la sauvegarde des modifications.
	Mapping links display as	**Auto** : par défaut, les liens sont en forme de courbes. **Curves** : les liens du mapping sont en forme de courbes. **Lines** : les liens du mapping sont en forme de lignes droites. Cette dernière option améliore légèrement les performances.
	Map editor	Le Mapper est l'éditeur du **tMap**. Il vous permet de définir les propriétés d'aiguillage et de transformation des données. Si vous ne souhaitez pas gérer les erreurs d'exécution, vous pouvez utiliser l'option **Die on error** (cochée par défaut), en haut à droite de l'éditeur. Cette option arrête le Job en cas d'erreur. Ce composant offre la fonction de schéma dynamique. Cela vous permet de récupérer des colonnes inconnues de fichiers source ou de copier des lots de colonnes d'une source sans avoir à mapper chaque colonne individuellement. Pour plus d'informations concernant les schémas dynamiques, consultez .*Onglet Dynamic settings* du **Guide Utilisateur Talend Integration Suite Studio**.
Utilisation	Plusieurs utilisations sont possibles, de la simple réorganisation des champs de données aux transformations les plus complexes, telles que le multiplexage et le démultiplexage de données, la concaténation, l'inversion, le filtrage, etc.	
Limitation	L'utilisation du composant **tMap** requiert un niveau moyen de connaissances Perl ou Java afin d'exploiter au mieux ses fonctionnalités. Ce composant est utilisé comme une étape intermédiaire, c'est pourquoi il ne peut pas être utilisé comme composant de début ou de fin de Job.	

Pour plus d'informations, consultez *Mapper des flux de données* du **Guide utilisateur** de **Talend Open Studio**.

Scénario 1 : Mapping simple avec filtre et jointure explicite

Le Job Java ci-dessous a pour objectif de lire des données d'un fichier stocké dans le **Repository**, de rechercher des données d'un fichier de référence distant puis d'extraire des données de ces deux fichiers en fonction de filtres et d'envoyer ces données vers le fichier de sortie et de rejet.

- Cliquez sur **File** dans la **Palette** de composants, sélectionnez **tFileInputDelimited** et déposez-le dans l'espace de modélisation. Changez le libellé en *Cars* (véhicules), soit en double-cliquant sur le libellé directement dans l'espace de modélisation graphique, soit via l'onglet **View** de la vue **Component**.
- Répétez cette opération et renommez le second composant d'entrée en *Owners* (propriétaires).
- Cliquez sur **Processing** dans la **Palette** de composants, sélectionnez le **tMap** et déposez-le dans l'espace de modélisation.
- Reliez les deux composants d'entrée, *Owners* et *Cars* au composant **tMap** et renommez les libellés des connexions respectives.

- Les métadonnées des fichiers délimités *Cars* et *Owners* sont définies dans le gestionnaire **Metadata** du **Repository**. Ainsi vous pouvez leur assigner les propriétés stockées dans le Repository dans la vue **Component**.
- Double-cliquez sur *Cars* pour paramétrer les propriétés.

- Sélectionnez **Repository** dans les champs **Property type** et **Schema type**. Le reste des champs est automatiquement renseigné lorsque vous sélectionnez la métadonnée appropriée dans la liste.
- Double-cliquez sur le composant *Owners* et définissez à nouveau les propriétés. Sélectionnez la métadonnée appropriée si elle existe. Sinon définissez manuellement chacune des propriétés.

Pour plus d'informations concernant la création de métadonnées dans le **Repository**, consultez *Centraliser les métadonnées* du **Guide utilisateur** de **Talend Open Studio**.

- Puis double-cliquez sur le composant **tMap** pour ouvrir le Mapper. La zone Input (données en entrée) est déjà renseignée avec les schémas des composants d'entrée et la première table correspond au flux principal (**Main Row**).
- Les libellés de chacune des connexions d'entrée apparaissent en en-tête des tables.
- Créez une jointure entre les deux tables d'entrée en faisant simplement glisser le champ *ID_Owner* de la table principale vers le champ équivalent de la table de référence.

- Définissez le lien en **Inner Join** en cliquant sur le bouton **tMap settings** et en sélectionnant **Join** pour *Join Model*.
- Cliquez sur le signe **[+]** dans la zone **Output** (sortie) du Mapper et ajoutez trois tables de sortie : *Insured*, *Reject_NoInsur*, *Reject_OwnerID*.

- Glissez et déposez le contenu de la table principale d'entrée vers la première table de sortie. Pour plus d'informations concernant le mapping de données, consultez *Mapper des flux de données* du **Guide utilisateur** de **Talend Open Studio**.
- Cliquez sur la flèche dotée d'un [+] pour ajouter une ligne de filtre. La table *Insured* rassemble les données des tables *Cars* et *Owners* qui possèdent un *ID Insurance*.

- Par conséquent, glissez-déposez le champ *ID_Insurance* vers la zone de filtre qui vient d'être créée et saisissez la formule d'exclusion des valeurs non définies suivante :
  `owners_data.ID_Insurance = null`
- La table *Reject_NoInsur* est un flux de rejet standard regroupant les données qui ne satisfont pas la condition de filtre définie. Cliquez sur la flèche orange pour définir cette table comme table de rejet standard.

- La troisième et dernière table est une table de rejets spécifiques pour les jointures Inner Join qui ne peuvent être établies. Il s'agit par exemple du cas où un champ *Owners_ID* de la table *Cars* ne correspond à aucune entrée du fichier *Owners*.
- Cliquez sur le bouton **tMap Settings** puis sélectionnez true pour *Catch lookup inner join reject* afin de définir la dernière table comme étant la table du flux de sortie de rejets Inner Join.
- Cliquez sur **OK** pour valider et revenir à l'espace de modélisation.
- Ajoutez trois composants **tFileOutputDelimited** au Job et reliez-les au Job à partir du composant **tMap** à l'aide d'un lien **Row** > **Main**.
- Réutilisez les trois libellés définis précédemment dans le mapper pour renommer vos connexions.

- Puis double-cliquez sur chacun des composants de sortie afin de définir leur chemin d'accès respectif. Si vous souhaitez créer un nouveau fichier, parcourez votre système jusqu'au dossier de destination et saisissez le nom du fichier avec son extension.

- Cochez la case **Include header** pour réutiliser les libellés des colonnes du schéma d'entrée dans votre fichier de sortie.

- Appuyez sur **F6** pour exécuter le Job ou passez à l'onglet **Run** et cliquez sur le bouton **Run**.
- Les fichiers de sortie sont créés.

Pour des exemples d'utilisation des schémas dynamiques avec le **tMap**, consultez :

- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478.*
- *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507.*

Scénario 2 : Mapping avec rejet Inner join

Ce scénario, basé sur le scénario 1, requiert l'ajout d'un fichier d'entrée, *Resellers*, contenant les informations sur les revendeurs, ainsi que l'ajout de champs supplémentaires dans le fichier de sortie principal. De plus, deux filtres sont ajoutés sur les jointures Inner Join afin de récupérer les rejets spécifiques.

- Cliquez sur **File** dans la **Palette** des composants et déposez un composant **tFileInputDelimited** dans l'espace de modélisation graphique.

- Connectez ce nouveau composant au mapper et ajoutez un libellé au composant et à cette connexion : ici, *Resellers*.

- Double-cliquez sur le composant *Resellers* pour définir les propriétés d'entrée.
- Parcourez votre système de fichier jusqu'au *Resellers.csv*. Editez le schéma et ajoutez les colonnes pour refléter la structure du fichier d'entrée.

- Vous pouvez également créer une métadonnée contenant la description de ce fichier et avoir la possibilité de réutiliser ses propriétés et son schéma en sélectionnant **Repository** dans les champs adéquats. Pour plus d'informations, consultez *Configurer un schéma File Delimited* du **Guide utilisateur** de **Talend Open Studio**.
- Double-cliquez sur le composant **tMap** et vous remarquerez que le schéma est automatiquement ajouté à la zone Input des schémas d'entrée.

- Créez une jointure entre le flux d'entrée principal (**Row Main**) et l'entrée de référence (**Lookup**), *Resellers*. Cochez la case **Inner Join** pour définir le type de jointure créé.
- Glissez-déposez les champs de la table *Resellers* vers la table de sortie principale.

Lorsque deux Inner Joins sont définis, vous devez soit définir deux tables de rejet inner join pour différencier les deux rejets, soit rassembler les deux rejets inner join dans une seule et même table de sortie, s'il n'y a qu'une seule sortie.

- Dans la zone de sortie Output, cliquez sur le signe **[+]** pour ajouter une nouvelle table de sortie.
- Donnez un nom à cette nouvelle connexion, *Reject_ResellerID*.
- Cliquez sur le bouton **tMap Settings** puis sélectionnez true pour *Catch lookup inner join reject* afin de définir la dernière table comme étant la table du flux de sortie de rejets Inner Join.
- Glissez-déposez deux champs de la table d'entrée principale (*Cars*) dans cette nouvelle table. Par conséquent, dans cet exemple, si la jointure Inner Join ne peut être établie pour une entrée, les données (*ID_Cars* & *ID_resellers*) seront rejetées dans le fichier de sortie approprié et permettront d'identifier rapidement le goulot d'étranglement.

- Appliquez maintenant un filtre sur chacune des sorties de rejet Inner Join afin de distinguer les deux types de rejets.
- Dans la première table de rejet (*Reject_OwnerID*), cliquez sur la flèche dotée d'un signe [+] pour ajouter une ligne de filtre et renseignez ce champ avec la formule suivante pour ne recueillir que les rejets liés au champ *OwnerID* non défini : `Owners_data.ID_Owner==null`
- Dans la seconde table de rejet Inner Join (*Reject_ResellerID*), renouvelez l'opération et utilisez la formule suivante : `Resellers_data.ID_Reseller==null`

- Cliquez sur **Apply** pour enregistrer les changements, puis sur **OK** pour fermer le Mapper.
- Cliquez-droit sur le composant **tMap**, cliquez sur **Row** et sélectionnez *Reject_ResellerID* dans la liste.
- Connectez le flux principal du Mapper au composant de sortie *Reseller*.

- Dans ce scénario, retirez du fichier *Resellers.csv* les lignes correspondant aux *Reseller ID 5 et 8*.
- Exécutez le Job via la touche **F6** du clavier, ou à partir de la vue **Run**.

- Les quatre fichiers de sortie sont tous créés dans le dossier défini dans les propriétés de la vue **Component**.
- Remarquez que dans le fichier de rejet lié à l'Inner Join, *NoResellerID.csv*, le champ *ID_Owners* correspond aux entrées du fichier *Cars* dont le Reseller ID est égal à 5 et 8 tel que défini précédemment.

Pour des exemples d'utilisation des schémas dynamiques avec le **tMap**, consultez :

- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478*.
- *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507*.

Scénario 3 : Mapping de jointure en cascade

Un troisième scénario plus avancé et basé sur le scénario 2 requiert l'ajout d'une nouvelle entrée contenant les informations d'assurance, par exemple.

Dans le **Mapper**, mettez en place une jointure **Inner Join** entre deux tables de référence (Lookup) des flux secondaires *Owners* et *Insurance* afin de gérer les références en cascade et par conséquent de récupérer les informations de la table *Insurance* via les données de la table *Owners*.

Scénario 4 : Mapping avancé avec filtres, jointures explicites et rejet Inner join

Ce scénario présente un Job utilisant le **tMap** et permettant de trouver les clients potentiels d'un *Reseller* (revendeur). Ces clients doivent être propriétaires d'une marque définie et avoir entre 2 et 6 enfants (inclus) dans un but de vente incitative.

- Cliquez-déposez les composants suivants de la Palette : **tFileInputDelimited** (x3), **tMap**, **tFileOutputDelimited** (x2)
- Connectez les composants d'entrée au tMap à l'aide de connexions de type **Main row**. Faites attention au premier composant que vous connectez car son flux de données sera automatiquement un flux **Main** (principal). Et toutes les autres connexions seront alors de type **Lookup** (flux de référence).
- Définissez les propriétés de chaque composant d'entrée dans l'onglet **Basic settings**. Par exemple, sélectionnez le chemin d'accès au fichier *Resellers* utilisé comme flux principal dans votre Job.

- Sélectionnez le fichier délimité à utiliser, les séparateurs de lignes et de champs (**Row** et **Field Separator**), les lignes d'en-tête (**Header**) et de bas de page (**Footer**), si nécessaire.
- Cliquez sur **Edit Schema**, si le schéma n'est pas stocké dans le **Repository**. Vous retrouverez ce schéma dans la table principale en haut de la zone **d'entrée Input** du Mapper.
- Répétez ces étapes pour les autres composants d'entrée : *Cars* et *Owners*. Ces deux flux de référence (**Lookup**) renseigneront les tables secondaires (lookup) de la zone d'entrée du Mapper.
- Cliquez sur le composant **tMap** pour ouvrir le Mapper et définir le mapping et les filtres.

- Tout d'abord, configurez les jointures explicites entre le flux **Main** et le flux **Lookup**.
- D'un simple glisser-déposer, déplacez la colonne *ID_Resellers* vers les colonnes correspondantes puis renseignez le champ **Expression key** dans la table **Lookup**.

- La jointure explicite apparaît en couleur tout comme la clé de hachage.
- Puis dans le champ **Expr. Key** de la colonne *Make* (Marque), saisissez (en Java) le filtre à appliquer. Dans ce cas d'utilisation, saisissez simplement *"BMW"* puisque ce qui est recherché ce sont les propriétaires (*Owners*) de cette marque en particulier.
- Effectuez une jointure en cascade entre les deux tables de référence *Cars* et *Owners*, afin de retrouver les informations sur les propriétaires et plus précisément le nombre d'enfants qu'ils ont.

- D'un simple glisser-déposer, déplacez la colonne *ID_Owners* de la table *Cars* vers le champ **Expr. Key** de la colonne *id_owner* de la table *Owners*.
- Cliquez sur le bouton **Filter** près de la case **Inner Join** pour afficher la zone de saisie du Filtre.
- Saisissez la déclaration de **Filtre** pour réduire le nombre de lignes prises en compte dans le flux **Lookup**. Dans cet exemple, la déclaration est la suivante : `Owners.Children_Nr > 1 && Owners.Children_Nr < 6`
- Comme vous souhaitez exclure les valeurs nulles dans une table séparée et les exclure de la sortie standard, cliquez sur le bouton **tMap Settings** puis sélectionnez **Inner Join** comme modèle de jointure pour chaque table **Lookup** à filtrer.

- Dans les paramètres du **tMap**, vous pouvez sélectionner une des options suivantes : **Unique match**, **First match**, ou **All Matches**. Dans cet exemple, sélectionnez l'option **All matches**. Ainsi, si plusieurs correspondances sont trouvées dans l'Inner Join (i.e. les lignes répondant à la jointure explicite ainsi qu'au filtre), elles seront toutes prises en compte dans le flux de sortie (soit dans le flux de rejet, soit dans le flux de sortie standard).

  L'option **Unique match** fonctionne de la même manière que l'option **Last match**. Les options **First match** et **All matches** fonctionnent comme l'indiquent leurs noms (respectivement première correspondance et toutes les correspondances).

- Puis dans la zone **Output** du **Mapper**, ajoutez deux tables : une pour les correspondances répondant à la jointure explicite et au filtre et une pour les rejets.
- Cliquez sur le bouton **[+]** pour ajouter des tables et donnez un nom aux flux de sortie.

- Glissez-déposez les données des tables **Main** et **Lookup** de la zone Input vers les tables de sortie respectives, suivant le type d'informations que vous souhaitez obtenir.
- Dans la table de rejet utilisée pour diriger les données ne répondant ni à la jointure externe ni au filtre, cliquez sur le bouton **tMap Settings** et sélectionnez **true** pour *Catch lookup inner join reject*.
- Dans l'éditeur graphique, cliquez droit sur le **tMap** et créez une connexion de sortie vers le composant correspondant.
- Définissez les paramètres des composants de sortie dans l'onglet **Basic settings**.

- Définissez le chemin d'accès, les séparateurs de lignes (**Row**) et de champs (**Field**), si nécessaire. Et pour cet exemple, cochez la case **Include Header**.

- Le **Schéma** devrait se propager automatiquement à l'aide du **Mapper**.
- Enregistrez votre Job, puis cliquez sur l'onglet **Run** et cochez la case **Statistics** pour suivre l'exécution du Job.

Les statistiques montrent que plusieurs correspondances ont été trouvées et ainsi la somme des lignes de sortie (Main + Rejet) est supérieure au nombre de lignes du flux d'entrée principal.

Pour des exemples d'utilisation des schémas dynamiques avec le **tMap**, consultez :

- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478.*
- *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507.*

Scénario 5 : Mapping avancé avec filtres et vérification des lignes

Ce scénario est une version modifiée du scénario précédent. Il décrit un Job qui applique des filtres et vérifie chaque ligne d'une entrée de référence.

- Reprenez le Job du *Scénario 4 : Mapping avancé avec filtres, jointures explicites et rejet Inner join, page 1305*.
- Aucune modification n'a été apportée aux fichiers Input delimited.
- Lancez le Mapper pour modifier le mapping et les filtres.

- Supprimez toutes les jointures explicites entre la table principale et la table de référence, *Cars*.
- Notez que l'option **All Matches** se change automatiquement en **All Rows**. En fait, comme aucune jointure explicite n'est déclarée (il n'y a pas de clé de hachage), toutes les lignes du flux de référence doivent être prises en compte et vérifiées en fonction des lignes du flux principal.
- Supprimez le filtre (*"BMW"*) du champ **Expr. key** de la table *Cars*.

- Cliquez sur le bouton **Filter** pour afficher la zone de saisie du filtre. Puis saisissez le nouveau filtre pour réduire la recherche aux marques *BMW* ou *Mercedes*. La déclaration du filtre est la suivante :
  `Cars.Make.equals("BMW") || Cars.Make.equals("Mercedes")`
- Le filtre appliqué sur la table de référence *Owners* ne change pas par rapport au scénario précédent.
- Définissez les nouveaux chemins d'accès pour les flux de sortie respectifs.
- Enregistrez le Job et activez les **Statistiques** dans l'onglet **Run** avant d'exécuter le Job.

Les Statistiques montrent qu'un produit cartésien a été effectué entre les lignes du flux principal avec les lignes filtrées du flux de référence.

BMWMercedes_wthChildren.csv

```
34  Cars & Pickup Specialist;5952;76;2251 JG 82;Mercedes;5
35  Cars & Pickup Specialist;5952;79;2930 CP 77;Mercedes;5
36  Cars & Pickup Specialist;5952;96;8506 ZQ 08;BMW;4
37  Cars & Pickup Specialist;5952;97;7757 KQ 65;BMW;5
38  Cars & Pickup Specialist;5952;99;9162 MC 60;Mercedes;4
39  Cars & Pickup Specialist;5952;100;0146 DA 20;BMW;5
40  Quality Car Resale;7794;9;9939 CJ 88;Mercedes;3
41  Quality Car Resale;7794;15;4563 ZB 33;BMW;3
42  Quality Car Resale;7794;20;3408 EW 35;Mercedes;2
43  Quality Car Resale;7794;27;5792 QT 18;BMW;3
44  Quality Car Resale;7794;33;8253 DP 32;BMW;3
45  Quality Car Resale;7794;44;3748 NN 21;BMW;2
46  Quality Car Resale;7794;45;4065 EA 69;Mercedes;3
```

Le contenu du flux de sortie principal montre que les lignes filtrées sont bien passées.

MWMercedes_wthChildren.csv | BMWMercedes_wthchildren_InnerReject.csv

```
1  Name_Reseller;ZipCode;ID_Owners;Reg_Car;Make;ID_Reseller;id_owner;name;
   id_insurance;Children_Nr
2  Cars & Pickup Shop;5113;16;6709 YE 10;BMW;27;16;bobouh;QYW1412;1
3  Cars & Pickup Shop;5113;37;5898 EB 09;BMW;54;37;hirtgall;MFR4898;1
4  Cars & Pickup Shop;5113;65;0439 XF 39;BMW;32;65;mauvaes;FBG6516;6
5  Cars & Pickup Shop;5113;68;8147 RS 83;Mercedes;33;68;bouhle;NGT4401;6
6  Cars & Pickup Shop;5113;80;1359 DY 17;Mercedes;8;80;gallgall;GAC9240;6
7  Cars & Pickup Shop;5113;86;0094 SH 41;BMW;26;86;otbo;OFU7978;6
8  Cars & Pickup Shop;5113;92;6544 LF 76;BMW;50;92;otmau;XNH2512;6
```

Alors que le résultat des rejets met en évidence les lignes qui ne correspondent à aucun des filtres.

Pour des exemples d'utilisation des schémas dynamiques avec le **tMap**, consultez :

- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478*.
- *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507*.

Scénario 6 : Mapping avancé avec rechargement de la base de référence à chaque ligne

Le scénario suivant décrit un Job permettant de récupérer certaines informations sur des personnes dans une base de données de référence, en se basant sur une jointure effectuée sur leur âge. Le flux principal des données source est lu à partir d'une table Mysql appelée *people_age* contenant certains détails sur les personnes, comme leur identifiant (numérique), leur nom et prénom (alphanumérique) ainsi que leur âge (numérique), qui peut être soit 40, soit 60 ans. On a intentionnellement réduit le nombre d'entrées de cette table.

Les informations de référence sont également stockées dans une table Mysql appelée *large_data_volume*. Cette table de référence contient un certain nombre d'entrées, dont le nom des villes dans lesquelles les personnes enregistrées dans la table principale sont allées. Le nombre d'entrées a été restreint pour plus de clarté mais, dans le cas d'une utilisation normale, le Job ci-dessous a des résultats plus significatifs quand on traite un gros volume de données de référence.

Pour optimiser les performances, un composant de connexion aux bases de données est utilisé au début de ce Job pour ouvrir une connexion à la base de données de référence afin de ne pas ouvrir et refermer une connexion à chaque ligne de la table de référence chargée.

Un filtre d'expression (expression filter) est appliqué à ce flux de référence, qui permet de ne sélectionner que les données des personnes dont l'âge est 60 ou 40 ans. Ainsi, seules les lignes pertinentes de la table de référence sont enregistrées pour chaque ligne du flux principal.

Ce Job présente donc comment, à partir d'un nombre limité de lignes de flux principal, on peut optimiser la base de données de référence pour n'enregistrer que les résultats correspondant à la clé d'expression (expression key).

En général, comme le chargement des références s'effectue pour chacune des lignes de flux principal, il est plus intéressant d'utiliser cette option quand on ne traite qu'un nombre limité de lignes de flux principal et un grand nombre de lignes de référence.

La jointure s'effectue au niveau du champ *age*. Les informations de la base de référence sont ensuite chargées pour chacune des lignes du flux principal en entrée, en utilisant l'option de chargement appropriée dans l'éditeur du composant **tMap**.

Dans cet exemple, les métadonnées ont été préparées pour le composant de connexion et les composants source. Pour plus d'informations sur les métadonnées de connexion aux bases de données et les métadonnées de création de schémas des tables, consultez la section dédiée au sujet dans le **Guide Utilisateur** de **Talend Open Studio**.

Le Job est formé de cinq composants, quatre composants de bases de données et un composant de mapping.

- A partir du noeud **Metadata** > **Db Connections** dans le **Repository**, glissez le schéma de base de données dans l'éditeur graphique. Dans cet exemple, la table source s'appelle *people_age*.
- Lorsque vous déposez le composant, une liste apparaît à l'écran : sélectionnez le composant **tMysqlInput**.

- Déposez le schéma de base de donnée de référence à partir du noeud **Metadata** > **Db Connections**. Dans cet exemple, la table de référence s'appelle *large_data_volume*.
- De la même manière, glissez une connexion à la base de données du noeud Db Connections dans le Repository dans l'éditeur graphique. Dans la boîte de dialogue **[Components]**, sélectionnez le composant **tMysqlConnection**. Ce composant crée une connexion permanente à la base de données de référence afin d'éviter que la connexion ne se referme au chargement que chaque ligne de référence.
- A partir de la **Palette** située à droite dans l'éditeur graphique, cliquez-déposez le connecteur **tMap** (famille **Processing**) et le composant **tMysqlOutput** (famille **Database**).
- Reliez maintenant tous les composants entre eux : faites un clic-droit sur le **tMysqlInput** correspondant à la table *people* et maintenez le bouton enfoncé tout en déplaçant le curseur jusqu'au **tMap**.
- Relâchez le bouton lorsque vous êtes sur le **tMap**, le flux principal de lignes est créé automatiquement.
- Renommez le lien de type **Main** en *people*, pour pouvoir identifier plus facilement le flux de données principal.
- Effectuez la même opération pour connecter la table de référence (*large_data_volume*) au **tMap** et le **tMap** au **tMysqlOutput**.

- Une boîte de dialogue apparaît et vous devez donner un nom au lien de sortie. Dans cet exemple, le flux de sortie s'appelle *people_mixandmatch*.
- Renommez également le lien de connexion à la base de référence en *large_volume* pour pouvoir identifier plus facilement le flux de données.
- Connectez le **tMysqlConnection** au **tMysqlInput** via un lien de type **OnSubjobOk**.
- Double-cliquez ensuite sur le composant **tMap** pour ouvrir l'éditeur graphique de mapping.

- La table **Output** (qui s'est créée automatiquement quand vous avez lié le **tMap** au **tMysqlOutput**) sera composée des lignes concordantes du flux de référence (*large_data_volume*) et du flux principal (*people_age*).
- Sélectionnez les lignes du flux principal que vous voulez passer en sortie et faites les glisser vers la table Output (dans le coin supérieur droit de l'éditeur de mapping) : ils y seront copiés.
- Dans cet exemple, la sélection des lignes du flux principal inclut les champs suivants : *id*, *first_name*, *last_Name* et *age*.
- Dans la table de référence, la colonne *city* est sélectionnée.
- Faites glisser les colonnes sélectionnées des tables d'entrée (*people* et *large_volume*) vers la table de sortie.
- Créez maintenant la jointure entre le flux principal et le flux de référence.

- Sélectionnez la colonne *age* dans la table du flux principal (en haut) et faites-la glisser vers la colonne *age* de la table du flux de référence (*large_volume* dans cet exemple).
- L'icône d'une clé apparaît à côté de l'expression liée dans la table de référence. La jointure est maintenant établie.
- Cliquez sur le bouton **tMap Settings** et sélectionnez l'option **Reload at each row** pour que la base de référence soit chargée à nouveau pour chaque nouvelle ligne traitée.

- Dans cet exemple vous devez également sélectionner l'option **All matches** dans la table de référence pour récupérer dans le flux de sortie toutes les occurrences de *age* qui correspondent.
- Implémentez ensuite le filtre, établi à partir de la colonne *age*, dans la table de référence. Le champ **GlobalMapKey** est créé automatiquement lorsque vous sélectionnez l'option **Reload at each row**. Vous pouvez donc utiliser cette expression pour filtrer les données de référence de manière dynamique, pour que seules les informations pertinentes soient chargées avec le flux principal.

Comme l'a souligné l'introduction de ce scénario, le flux de données principal contient uniquement les personnes dont l'âge vaut 40 ou 60. Pour éviter de charger toutes les lignes de la table de référence, dont les âges sont différents de 40 et 60, vous pouvez utiliser la valeur âge du flux principal comme variable globale pour alimenter le filtre de référence.

- Glissez la colonne *Age* de la table de flux principal vers le champ **Expr.** dans la table de référence.
- Renseignez ensuite le nom de la variable dans le champ **globalMap Key**, en utilisant une expression. Dans cet exemple, l'expression est : `"people.Age"`.
- Cliquez sur le bouton **OK** pour enregistrer les paramètres de mapping et retourner dans l'éditeur graphique.
- Pour terminer l'implémentation du filtrage dynamique du flux de référence, vous devez maintenant ajouter une clause "where" à la requête de la base de données d'entrée.

- A la fin du champ **Query**, juste après la commande Select, entrez la clause de type "where" suivante : `WHERE AGE = " + ((Integer)globalMap.get("people.Age"))`.

- Assurez-vous que le type correspond à la colonne utilisée comme variable. Dans cet exemple, la donnée *Age* est du type `Integer`. Veillez aussi à utiliser la variable de la manière dont vous l'avez paramétrée dans le champ **globalMap key** de l'éditeur de mapping.
- Double-cliquez ensuite sur le composant **tMysqlOutput** et vérifiez que le schéma correspond aux paramètres de mapping.
- Tout est maintenant prêt pour l'exécution du Job.
- Cliquez sur l'onglet **Run**, situé en bas de l'éditeur graphique, pour afficher la vue d'exécution du Job.

- Avant d'exécuter le Job, sélectionnez l'option **Traces** pour observer l'avancée du traitement des données.
- Assurez-vous que les types de données sont respectés (string, integer...) pour éviter les erreurs d'exécution.
- Appuyez ensuite sur la touche **F6** ou cliquez sur le bouton **Run** pour exécuter le Job.

Pour une meilleure visibilité, il est possible d'agrandir la vue du Job dans l'éditeur graphique lors de l'exécution en double-cliquant simplement sur l'onglet portant le nom du Job.

Les données de référence sont chargées à nouveau pour chaque ligne du flux principal qui correspond à la contrainte d'âge. Toutes les correspondances à la condition *age* sont récupérées à partir des lignes de la table de référence et regroupées dans le flux de sortie.

Si vous vérifiez les données contenues dans la table *people_mixandmatch* nouvellement créée, vous y trouverez toutes les copies basées sur *age* correspondant aux différentes personnes dont l'âge vaut 40 ou 60, ainsi que les villes où ils sont allés.

id	First_Name	Last_...	AGE	CITY
1	Rutherford	Polk	40	Montpelier
1	Rutherford	Polk	40	Austin
1	Rutherford	Polk	40	Providence
1	Rutherford	Polk	40	Lansing
1	Rutherford	Polk	40	Denver
1	Rutherford	Polk	40	Jefferson City
2	Harry	Tyler	40	Montpelier
2	Harry	Tyler	40	Austin
2	Harry	Tyler	40	Providence
2	Harry	Tyler	40	Lansing
2	Harry	Tyler	40	Denver
2	Harry	Tyler	40	Jefferson City
3	Martin	John...	60	Harrisburg
4	Rutherford	Taylor	40	Montpelier

Pour des exemples d'utilisation des schémas dynamiques avec le **tMap**, consultez :

- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478.*
- *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507.*

Scénario 7 : Mapping avec des tables de sortie

Le scénario suivant décrit un Job traitant des flux de rejet sans les séparer du flux principal.

- Dans la vue **Repository**, cliquez sur **Metadata** > **File delimited**. Glissez-déposez la métadonnée *customers* dans l'espace de modélisation graphique. Pour plus d'informations concernant la centralisation des métadonnées, consultez *Centraliser les métadonnées, page 145* du **Guide Utilisateur** **Talend Open Studio**. La métadonnée *customers* contient des informations relatives aux clients, comme leur ID, leur nom, leur adresse, etc.
- Une boîte de dialogue s'ouvre et vous demande de choisir le type de composant que vous souhaitez utiliser.Sélectionnez le composant **tFileInputDelimited** et cliquez sur **OK**.
- Ensuite, glissez-déposez la métadonnée *states*. Sélectionnez le même composant dans la boîte de dialogue, et cliquez sur **OK**. La métadonnée *states* contient l'identifiant de l'état, ainsi que son nom.
- Glissez-déposez de la **Palette** un **tMap** et deux composants **tLogRow**.
- Reliez le composant *customers* au **tMap** à l'aide d'un lien **Row Main**.
- Reliez le composant *states* au **tMap**, en utilisant un lien **Row Main**. Il sera automatiquement défini comme **Lookup**.
- Double-cliquez sur le composant **tMap**. Glissez-déposez la colonne *idState* dans la première ligne de la seconde table, afin de créer une jointure.
- Désélectionnez la case **Die on error** si vous souhaitez gérer les erreurs d'exécution. La table **ErrorReject** est maintenant créée. Cliquez quatre fois sur le bouton **[+]** en haut à droite de l'éditeur, afin d'ajouter quatre colonnes. Sélectionnez les colonnes d'entrée *id*, *idState*, *RegTime* et *RegisterTime* puis glissez et déposez-les dans les quatre nouvelles colonnes vides.

- Cliquez sur le bouton **[+]** en haut à droite de l'éditeur afin d'ajouter une table de sortie. Dans la boîte de dialogue qui s'ouvre, sélectionnez **New output**. Dans le champ correspondant, saisissez le nom de la table, *out1*. Cliquez sur **OK**.
- Ensuite, glissez-déposez les quatre colonnes suivantes de la table d'entrée dans la table *out1* : *id*, *CustomerName*, *CustomerAddress*, et *idState* et *labelState*.
- Ajoutez deux colonnes *RegTime* et *RegisterTime* avec un format Date, en cliquant deux fois sur le bouton **[+]**.
- Cliquez sur la première, et appuyez sur **Ctrl+Espace** pour afficher la liste d'autocomplétion. Sélectionnez `TalendDate.parseDate`. Changez le modèle en `("dd/MM/yyyy HH:mm",row1.RegTime)`.
- Répétez l'opération pour la seconde colonne, mais changez le modèle en `("yyyy-MM-dd HH:mm:ss.SSS",row1.RegisterTime)`.

- Ensuite, cliquez sur le bouton **[+]** en haut à droite de l'éditeur, afin d'ajouter une table de sortie. Dans la boîte de dialogue qui s'ouvre, sélectionnez **Create join table from**, choisissez *Out1*, et donnez le nom *rejectInner*. Cliquez sur **OK**.
- Cliquez sur le bouton **tMap Settings** et sélectionnez **true** pour *Catch lookup inner join reject*, afin de gérer les rejets.
- Cliquez sur la colonne *LabelState*, et appelez-la "UNKNOWN".
- Dans la colonne *RegTime*, appuyez sur **Ctrl+Espace**, et sélectionnez `TalendDate.parseDate`. Changez le modèle en `("dd/MM/yyyy HH:mm",row1.RegTime.)`
- Dans la colonne *RegisterTime*, appuyez sur **Ctrl+Espace**, et sélectionnez `TalendDate.parseDate`, mais changez le modèle en `("yyyy-MM-dd HH:mm:ss.SSS",row1.RegisterTime)`.
- Si les données de la ligne *row1* a un modèle incorrect, elles seront retournées par le flux *ErrorReject*.
- Cliquez sur **OK** afin de valider les modifications et fermer l'éditeur.
- Double-cliquez sur le composant **tLogRow** afin d'afficher sa vue **Component**.
- Dans la zone **Mode**, sélectionnez **Table.**
- Répétez l'opération avec le second **tLogRow**.
- Ensuite, sauvegardez votre Job et appuyez sur **F6** pour l'exécuter.

```
-----------------------------+---------------------------+-------+-----------
                                                tLogRow_2
-----------------------------+---------------------------+-------+-----------
CustomerName                 |CustomerAddress            |idState|LabelState
-----------------------------+---------------------------+-------+-----------
Griffith Paving and Sealcoatin|talend@apres91            |7      |Connecticut
Bill's Dive Shop             |511 Maple Ave.  Apt. 1B    |35     |Ohio
Childress Child Day Care     |662 Lyons Circle           |1      |Alabama
Facelift Kitchen and Bath    |unknown                    |0      |UNKNOWN
Terrinni & Son Auto and Truck |770 Exmoor Rd.            |0      |UNKNOWN   |
Kermit the Pet Shop          |1860 Parkside Ln.          |28     |Nevada
Tub's Furniture Store        |807 Old Trail Rd.          |15     |Iowa
```

La **Console** affiche les flux principaux et de rejet, traités par le composant **tMap**, et unifiés en un même flux de sortie.

Pour des exemples d'utilisation des schémas dynamiques avec le **tMap**, consultez :

- *Scénario : Ecrire des colonnes dynamiques d'une base de données MySQL vers un fichier de sortie, page 478.*
- *Scénario 4 : Ecrire des colonnes dynamiques d'un fichier source vers une base de données MySQL, page 507.*

tNormalize

Propriétés du tNormalize

Famille de composant	Processing/Fields	
Fonction	Le composant **tNormalize** normalise un flux entrant en fonction du standard SQL.	
Objectif	Le **tNormalize** permet d'améliorer la qualité de données et donc facilite la mise à jour des données.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Pour ce composant, le schéma est en lecture seule.
		Built-in : Le schéma est créé et conservé pour ce composant uniquement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
	Column to normalize	Sélectionnez la colonne du flux entrant sur laquelle est basé la normalisation.
	Item separator	Renseignez le séparateur délimitant les données du flux entrant.
Advanced settings		
Java uniquement	*Get rid of duplicated rows from output*	Cochez cette case pour supprimer les lignes doublons du flux de sortie.
Java uniquement	*Use CSV parameters*	Cochez cette case pour prendre en compte les paramètres spécifiques aux fichiers CSV, notamment la manière de protéger les caractères dans le champ **escape mode** et le type de guillemet dans le champ **enclosure**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant intermédiaire dans un flux de données.	
Limitation	n/a	

Scénario : Normaliser des données

Ce scénario illustre un Job qui normalise une liste de mots-clés des sujets d'un forum puis les transmet sous forme de tableau dans la console de la vue **Run**.

- Cliquez et déposez les composants suivants dans l'espace de modélisation : **tFileInputDelimited**, **tNormalize**, **tLogRow**.
- Dans l'onglet **Basic settings** du composant **tFileInputDelimited**, sélectionnez le fichier à normaliser.

- Le schéma du fichier est stocké dans le Repository. Il est composé d'une colonne, appelée *Tags*, et chaque ligne possède un ou plusieurs mots-clés.
- Dans les champs **Row Separator** et **Field Separator**, renseignez les séparateurs de lignes et de champs.

- Dans l'onglet **Basic settings** du composant **tNormalize**, dans la liste **Column to normalize**, sélectionnez la colonne sur laquelle la normalisation est basée.
- Dans ce scénario, la colonne à normaliser est *Tags*.

- Dans le champ **Item separator**, renseignez le séparateur de données, ici la virgule, que vous devez saisir entre guillemets.
- Dans les propriétés du composant **tLogRow**, sélectionnez le mode **Table (print values in the cells of table)**.
- Enregistrez le Job et exécutez-le, en appuyant sur **F6**.

```
Starting job tNormalize at 16:13 09/03/2010.

[statistics] connecting to socket on port 3820
[statistics] connected
.-------------------.
|     tLogRow_1     |
|=------------------=|
|Tags               |
|=------------------=|
|ldap               |
|db2                |
|jdbc driver        |
|grid computing     |
|talend architecture|
|context            |
|environment        |
|tmap               |
|eclipse            |
|database           |
|java               |
|postgresql         |
|tmap               |
|database           |
|java               |
|sybase             |
|deployment         |
|perl               |
|repository         |
|database           |
|informix           |
|java               |
'-------------------'
[statistics] disconnected
Job tNormalize ended at 16:13 09/03/2010. [exit code=0]
```

Les valeurs sont normalisées et affichées dans un tableau dans la console.

tPerl

Propriétés du tPerl

Famille du composant	Processing	
Fonction	Le **tPerl** transforme n'importe quelle donnée saisie en code Perl.	
Objectif	Le **tPerl** est un éditeur (Perl) et aussi un outil très flexible dans le Job.	
Basic settings	*Code*	Saisissez le code Perl en fonction de la commande ou de la tâche que vous souhaitez exécuter. Pour plus d'informations concernant la syntaxe des fonctions Perl, consultez l'aide en ligne de **Talend Open Studio** dans le menu **Talend Open Studio** > **User Guide** > Perl.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Ce composant est généralement utilisé pour débogger une erreur mais peut aussi être utlisé pour afficher le contenu d'une variable.	
Limitation	Ce composant est dédié aux utilisateurs Perl avancés et ne doit pas être utilisé avec un lien de type Row Main car il est destiné à être utilisé seul.	

Scénario : Afficher le nombre de lignes traitées

Ce scénario est un Job à trois composants qui affiche dans la console de log le nombre de lignes traitées et passées en sortie dans un fichier Excel.

- A partir de la **Palette**, cliquez-déposez les trois composants suivants dans l'éditeur graphique : le **tFileInputDelimited**, le **tFileOutputExcel** et le **tPerl**.
- Cliquez-droit sur le **tFileInputDelimited** et connectez-le au **tFileOutputExcel** à l'aide d'un lien de type **Row Main**.

- Cliquez-droit à nouveau sur le **tFileInputDelimited** et reliez-le au **tPerl** à l'aide d'un lien **Trigger** > **OnSubjobOk**. Ce lien signifie que, comme l'indique la flèche, le premier composant (**tFileDelimited**) s'exécutera avant le second (**tPerl**).
- Cliquez une fois sur le **tFileInputDelimited** et sélectionnez l'onglet **Basic settings** de la vue **Component** pour définir les propriétés du composant.

- Les propriétés (**Properties**) ne sont pas réutilisées pour ou à partir d'un Job stocké dans le **Repository**, elles sont au contraire utilisées pour ce Job uniquement. Par conséquent, sélectionnez l'option **Built-In** dans la liste déroulante du champ **Property Type**.
- Renseignez le chemin d'accès au fichier contenant les données à traiter ou parcourez vos dossiers jusqu'à celui-ci. Dans cet exemple, le fichier texte comprend une liste de noms associée aux adresses email correspondantes.
- Définissez les séparateurs de ligne (**Row**) et de champ (**Field**). Dans ce scénario, chaque ligne comprend un nom et son adresse email associée. Les champs sont séparés par des points-virgules.
- La première ligne du fichier contient le nom des colonnes, elle doit donc être ignorée dans ce Job. La valeur de l'en-tête (champ **Header**) est alors de 1.
- Il n'est pas nécessaire de définir un pied-de-page (**Footer**) ou une valeur maximale (**Limit**) pour ce scénario.
- Le champ **Schema type** est également défini en mode **Built-in** dans cet exemple. Cliquez sur **Edit Schema** pour décrire le contenu du fichier d'entrée. Dans ce scénario, il existe deux colonnes appelées *Name* et *Emails* contenant des chaînes de caractères (**String**) dont la longueur n'est pas définie. La clef est portée par le champ *Email*.
- Sélectionnez le composant **tFileOutputExcel** et définissez-en les paramètres en fonction de vos besoins.
- Sélectionnez le chemin d'accès au fichier de sortie, appelé *Sheet*, puis synchronisez le schéma.
- Définissez ensuite le sous-Job **tPerl** de manière à récupérer le nombre de lignes lues par le **tFileInputDelimited**.

- Entrez la commande Perl `print` pour afficher la variable relative au nombre de lignes lues dans le **tFileInputDelimited**. Pour accéder à la liste des variables disponibles, appuyez sur **Ctrl+Espace** puis sélectionnez la variable adéquate à partir de la liste.
- Pour une meilleure lisibilité de la console de log **Run Job**, ajoutez le signe égal avant et après la commande. Notez que les commandes, les chaînes de caractères et les variables apparaissent dans des couleurs différentes.
- Passez à la vue **Run Job** et exécutez le Job.

Le Job s'exécute normalement et crée un fichier Excel en sortie qui correspond au schéma à deux colonnes ayant été défini : *Name* et *Email*.

Le résultat de la commande Perl apparaît dans la console de log du Job.

tPivotToRows

Propriétés du tPivotToRows

Famille du composant	Processing	
Fonction	Le composant **tPivotToRows** transforme des colonnes en lignes.	
Objectif	Le composant **tPivotToRows** permet de choisir une liste de colonnes du flux entrant pour en fournir l'équivalent sous forme de lignes dans le flux sortant.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Row keys	Sélectionnez la liste des colonnes du schéma d'entrée que vous voulez afficher sous forme de ligne unique. Les colonnes non sélectionnées constitueront le pivot. Cliquez sur le bouton **[+]** pour ajouter autant de lignes que de colonnes à concaténer. Dans le champ **Input column** de chaque ligne, sélectionnez le nom des colonnes choisies.
	Row key concatenate delimiter	Déterminez le séparateur des valeurs concaténées.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est généralement utilisé comme composant intermédiaire dans un flux de données, il nécessite donc un composant en entrée et un composant en sortie.	

Scénario : Concaténer une liste de colonnes à partir d'une table en utilisant les autres colonnes comme pivot

Le scénario suivant est un Job Perl à quatre composants permettant de concaténer sur une même ligne les informations de plusieurs colonnes d'une table, en utilisant les autres colonnes comme pivot.

- A partir de la **Palette**, cliquez-déposez les composants suivants dans l'éditeur graphique : le **tFileInputDelimited**, le **tPivotToRows** et deux **tLogRow**.
- Connectez les quatre composants à l'aide de liens **Row Main** comme ci-dessus.
- Dans l'espace graphique, cliquez sur le composant **tFileInputDelimited**.
- Cliquez sur la vue **Component** pour configurer les paramètres de base (**Basic settings**) du **tFileInputDelimited**.

- Dans le champ **Property Type**, sélectionnez le mode **Repository** si vous avez stocké les informations du fichier délimité sous le noeud **Metadata** du **Repository** ou sélectionnez le mode **Built-In** pour les définir manuellement. Dans ce scénario, on utilise le mode **Built-In**.
- Renseignez le chemin d'accès au fichier à traiter dans le champ **File Name**. Dans cet exemple, on utilise le fichier *use_case_tunpivotrow*, qui comporte huit colonnes : *id*, *CustomerName*, *CustomerAddress*, *id2* et *RegisterTime* qui sont destinées à être concaténées, ainsi que *Sum1*, *Sum2* et *Sum3* qui serviront de pivot.
- Définissez si nécessaire les séparateurs de lignes (**Row separator**) et de champs (**Field separator**), l'en-tête (**Header**) et le pied-de-page (**Footer**), ainsi que le nombre de lignes à traiter (**Limit**). La première ligne, qui porte le nom des colonnes, est ignorée dans cet exemple, c'est pourquoi le champ **Header** est défini à "1".

- Dans le champ **Schema**, sélectionnez le mode **Built in** puis cliquez sur le bouton **[...]** à côté du champ **Edit Schema** pour définir les données à passer au composant suivant. Dans cet exemple, le schéma est constitué des huit colonnes du fichier d'entrée *use_case_tunpivotrow.*

Column	Key	Type	Nullable
id	☐	int	☑
CustomerName	☐	string	☑
CustomerAdress	☐	string	☑
id2	☐	int	☑
RegisterTime	☐	string	☑
Sum1	☐	int	☑
Sum2	☐	int	☑
Sum3	☐	int	☑

- Dans l'espace graphique, cliquez sur le composant **tPivotToRows**.
- Cliquez sur la vue **Component** pour configurer les paramètres de base (**Basic settings**) du **tPivotToRow**s.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent. Vous pouvez cliquer sur le bouton [...] à coté du champ **Edit schema** pour consulter le schéma.
- Cliquez sur le bouton **[+]** pour insérer dans la zone **Row keys** autant de lignes qu'il y a de colonnes à concaténer. Dans cet exemple, on a ajouté cinq lignes.

- A partir de la liste déroulante du champ **Input column**, sélectionnez pour chaque ligne l'intitulé de la colonne à concaténer. Les colonnes du schéma d'entrée qui ne seront pas sélectionnées serviront de pivot.
  Dans cet exemple, les colonnes *id*, *CustomerName*, *CustomerAddress*, *id2* et *RegisterTime* sont sélectionnées pour la concaténation alors que les colonnes *Sum1*, *Sum2* et *Sum3* sont laissées de côté pour servir de pivot.

- Dans le champ **Row key concatenate delimiter**, définissez un caractère pour séparer les données des différentes colonnes une fois la concaténation achevée.
- Cliquez sur un des composants **tLogRow** puis cliquez sur la vue **Component** pour configurer leurs paramètres de base (**Basic settings**).

- Choisissez le mode **Table** pour visualiser le fichier de départ et le résultat du **tPivotToRows** au même format et ainsi pouvoir les comparer.
- Utilisez les mêmes paramètres pour le second composant **tLogRow**.
- Enregistrez le Job puis appuyez sur **F6** pour l'exécuter.

```
Starting job test2 at 14:29 30/07/2009.

[stat] connecting to socket on port 4008 ...
[stat] connected
.-----------------------------------------------------------------------------------------------------------------------.
|                                                          tLogRow_1                                                    |
+----+-------------------------------+-----------------------+----+---------------------+----------+-----------+---------+
| id | CustomerName                  | CustomerAdress        | id2 | RegisterTime        | Sum1     | Sum2      | Sum3    |
+----+-------------------------------+-----------------------+----+---------------------+----------+-----------+---------+
|  1 | Griffith Paving and Sealcoatin | talend@apres91        |  41 | 2001-01-17 06:26:40.000 | 67852.0  | 61521.484 |  4949.0 |
|  2 | Bill's Dive Shop              | 511 Maple Ave.  Apt. 1B |   5 | 2002-06-07 09:40:00.000 | 88792.0  |   15434.1 |  6068.0 |
|  3 | Childress Child Day Care      | 662 Lyons Circle      |  28 | 1990-04-01 21:00:00.000 | 35340.0  |  17856.88 |  4029.0 |
|  4 | Facelift Kitchen and Bath     | 220 Vine Ave.         |  15 | 1972-04-23 18:00:00.000 |  6097.0  |  55560.24 | 10862.0 |
'----+-------------------------------+-----------------------+----+---------------------+----------+-----------+---------'
.-----------------------------------------------------------------------------------------------------.
|                                                   tLogRow_2                                         |
+-----------------------------------------------------------------------------------+-----------+-------------+
| row_key                                                                           | pivot_key | pivot_value |
+-----------------------------------------------------------------------------------+-----------+-------------+
| 1,Griffith Paving and Sealcoatin,talend@apres91,41,2001-01-17 06:26:40.000        | Sum1      |     67852.0 |
| 1,Griffith Paving and Sealcoatin,talend@apres91,41,2001-01-17 06:26:40.000        | Sum2      |   61521.484 |
| 1,Griffith Paving and Sealcoatin,talend@apres91,41,2001-01-17 06:26:40.000        | Sum3      |      4949.0 |
| 2,Bill's Dive Shop,511 Maple Ave.  Apt. 1B,5,2002-06-07 09:40:00.000              | Sum1      |     88792.0 |
| 2,Bill's Dive Shop,511 Maple Ave.  Apt. 1B,5,2002-06-07 09:40:00.000              | Sum2      |     15434.1 |
| 2,Bill's Dive Shop,511 Maple Ave.  Apt. 1B,5,2002-06-07 09:40:00.000              | Sum3      |      6068.0 |
| 3,Childress Child Day Care,662 Lyons Circle,28,1990-04-01 21:00:00.000            | Sum1      |     35340.0 |
| 3,Childress Child Day Care,662 Lyons Circle,28,1990-04-01 21:00:00.000            | Sum2      |    17856.88 |
| 3,Childress Child Day Care,662 Lyons Circle,28,1990-04-01 21:00:00.000            | Sum3      |      4029.0 |
| 4,Facelift Kitchen and Bath,220 Vine Ave.,15,1972-04-23 18:00:00.000              | Sum1      |      6097.0 |
| 4,Facelift Kitchen and Bath,220 Vine Ave.,15,1972-04-23 18:00:00.000              | Sum2      |    55560.24 |
| 4,Facelift Kitchen and Bath,220 Vine Ave.,15,1972-04-23 18:00:00.000              | Sum3      |     10862.0 |
'-----------------------------------------------------------------------------------+-----------+-------------'
Job test2 ended at 14:29 30/07/2009. [exit code=0]
```

La console de log présente les résultats des deux composants **tLogRow**. Le tableau tLogRow_1 donne un aperçu du fichier de départ et le tableau tLogRow_2 montre la concaténation des colonnes *id*, *CustomerName*, *CustomerAddress*, *id2* et *RegisterTime* ainsi que la transformation des colonnes *Sum1*, *Sum2* et *Sum3* en pivot.

tReplace

Propriétés du tReplace

Famille de composant	Processing	
Fonction	Le composant **tReplace** effectue un Rechercher/Remplacer dans les colonnes d'entrée spécifiées.	
Objectif	Ce composant permet de nettoyer tous les fichiers avant traitement.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Deux colonnes en lecture seule sont automatiquement ajoutées au schéma de sortie : Value et Match.
		Built-in : Le schéma est créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Simple Mode Search / Replace	Cliquez sur le bouton **[+]** pour ajouter autant de conditions que vous le souhaitez. Les conditions sont exécutées l'une après l'autre pour chaque ligne. **Input column** : Sélectionnez la colonne du schéma sur laquelle effectuer le Rechercher/Remplacer. **Search** : Saisissez la valeur à chercher dans la colonne d'entrée. **Replace with** : Saisissez la valeur de remplacement. **Whole word** : Cochez cette case si la valeur recherchée constitue un mot entier. **Case sensitive** : Cochez cette case pour tenir compte de la casse. Notez que vous ne pouvez pas utiliser d'expression régulière dans ces colonnes.
Advanced settings	*Use advanced mode*	Cochez cette case si vous souhaitez utiliser des expressions régulières.
Utilisation	Ce composant n'est pas un composant de début puisqu'il requiert un flux de données. Il requiert aussi un composant de sortie.	

Scénario : Remplacement mutliple et filtrage de colonnes

Le Job suivant recherche et remplace différentes coquilles et fautes dans un fichier .CSV puis filtre une colonne avant de générer en sortie un nouveau fichier .CSV contenant les données modifiées.

- Cliquez et déposez les composants suivants dans l'espace de modélisation : **tFileInputDelimited**, **tReplace**, **tFilterColumn** et **tFileOutputDelimited**.
- Connectez les composants à l'aide de liens de type **Main Row** via un clic-droit sur chaque composant.
- Sélectionnez le composant **tFileInputDelimited** et paramétrez les propriétés du flux d'entrée.

- Dans la liste déroulante **Property type**, sélectionnez l'option **Built-in**. Puis renseignez manuellement les champs suivants. Si les propriétés sont centralisées dans le Repository, choisissez l'option Repository et les champs seront automatiquement remplis.
- Le **Fichier** est un simple fichier .csv stocké localement. Ainsi, dans le champ **Filepath**, renseignez le chemin d'accès au fichier. Dans le champ **Row Separator**, renseignez le séparateur de lignes, ici le retour chariot. Dans le champ **Field Separator**, renseignez le séparateur de champs, ici un point virgule. Dans cet exemple, il n'y a pas d'en-tête (**Header**), de pied de page (**Footer**) ou de nombre limite de lignes à traiter (**Limit**).
- Le fichier contient des caractères tels que : `\t`, `|||`, `[d]` ou `*d` qui ne devraient pas être interprétés comme des caractères spéciaux ou comme des caractères génériques.

|||	Bill	Clinton	100
stre	John	Kennedy	98.30 $
Streat	Richar[d]	Nixon	78.23$
stret	Jimmy\t	Carter	38.54$
Streat	Richar+d	Nixon	78.2 $
Streat	toto	Nixon	78.23
Streat	Richar*d	Nixon	78.23$
street	Georges	bush	99.99$

- Dans le champ **Schema**, sélectionnez aussi l'option **Built-in**. Le schéma est composé de quatre colonnes de différents types : chaînes de caractères (string) ou entier (int).
- Sélectionnez le composant **tReplace** pour paramétrer les propriétés des Rechercher/Remplacer.

- Le schéma peut être synchronisé à partir du flux d'entrée.
- Cochez la case **Simple mode** car les paramètres de recherche peuvent être facilement définis sans avoir recours aux expressions régulières.
- Cliquez sur le signe **[+]** pour ajouter des lignes aux tableaux des paramètres.
- En premier paramètre, sélectionnez *amount* dans la colonne **Input column**. Dans la colonne **Search**, cherchez tous les *points* et dans la colonne **Replace** indiquez la valeur de remplacement, ici la *virgule*. Les valeurs doivent être mises entre guillemets.
- En deuxième paramètre, sélectionnez *str* dans la colonne **Input column**. Dans la colonne **Search**, cherchez les chaînes de caractères *stret* ou *streat* ou *stre*. Notez que ces valeurs sont séparées par une barre verticale qui signifie "OU" et qui équivaut au `or` en langage Perl. Dans la colonne **Replace**, renseignez la valeur de remplacement, ici *Street*. Cochez la case **whole word**.
- En troisième paramètre, sélectionnez à nouveau *str* dans la colonne **Input column**. Dans la colonne **Search**, cherchez les barres verticales en mettant une barre oblique inversée devant pour éviter de la confondre avec le "or" en langage Perl. Et dans la colonne **Replace**, ne mettez rien entre les guillemets.

- En quatrième paramètre, sélectionnez *firstname* dans la colonne **Input column**. Dans le champ **Search**, cherchez les caractères suivants : `[, ], +, *`. Notez que ces valeurs sont séparées par une barre verticale qui signifie “OU” et qui est équivaut au `or` en langage Perl. Et dans la colonne **Replace**, ne mettez rien entre les guillemets.
- En cinquième paramètre, sélectionnez *amount* dans la colonne **Input column**. Dans le champ **Search**, saisissez le signe du dollar entre guillemets simples et dans le champ **Replace**, saisissez le signe de l’euro entre guillemets.
- En dernier paramètre, sélectionnez *firstname* dans la colonne **Input column**. Dans la colonne **Search**, cherchez la chaîne `\t`. Pour différencier cette chaîne de la tabulation, ajoutez autant de barres obliques inversées devant qu’il y a de vérifications. En d’autres termes, il faut ajouter deux barres obliques inversées pour que la chaîne ne soit pas confondue avec la tabulation et deux autres qui font partie du caractère recherché. En tout, il doit y avoir quatre barres obliques inversées inclue celle constituant la chaîne de caractères. Dans le champ **Replace**, ne mettez rien entre les guillemets. Cochez la case **whole word**.
- L’option **Advanced mode** n’est pas utilisée pour ce scénario.
- Sélectionnez le composant suivant : **tFilterColumn**.

- Le composant **tFilterColumn** contient un éditeur de schéma permettant de créer un schéma de sortie basé sur le nom des colonnes du schéma d’entrée. Dans cet exemple, changez l’ordre des colonnes et ajoutez trois nouvelles colonnes pour obtenir le schéma suivant : *empty_field, firstname, name, str, amount, filler1, filler2*.
- Cliquez sur **OK** pour valider.
- Double-cliquez sur le **tFileOutputDelimited** afin d’afficher sa vue **Component**.

- Dans le champ **Property Type**, sélectionnez l'option **Built-in** et paramétrez les propriétés du composant **tFileOutputDelimited** manuellement.
- Dans le champ **Schema**, sélectionnez l'option **Built-in** et cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent.
- Enregistrez le Job et exécutez-le.

	Bill	Clinton		100
	John	Kennedy	Street	98,30 €
	Richard	Nixon	Street	78,23 €
	Jimmy	Carter	Street	38,54 €
	Richard	Nixon	Street	78,20 €
	toto	Nixon	Street	78,23
	Richard	Nixon	Street	78,23 €
	Georges	bush	street	99,99 €

La première colonne est vide et les caractères parasites ont été supprimés des autres colonnes. La colonne street a été déplacée, les décimales sont dorénavant séparées par une virgule et le symbole monétaire a été changé.

tSampleRow

Propriétés du tSampleRow

Famille du composant	Processing	
Fonction	Le composant **tSampleRow** filtre les lignes en fonction de leur position.	
Objectif	Le **tSampleRow** permet de sélectionner des lignes en fonction d'une liste de lignes simples et/ou de groupes de lignes.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Range	Renseignez la position ou l'intervalle choisi, en utilisant la syntaxe appropriée, pour récupérer une liste de lignes simples et/ou de groupes de lignes.
Utilisation	Ce composant traite des flux de données, il nécessite donc des composants d'entrée et sortie.	
Limitation	n/a	

Scénario : Filtrer des lignes et des groupes de lignes

Ce scénario Java décrit un Job à trois composants. On utilise un composant **tRowGenerator** pour créer des entrées au hasard qui sont directement envoyées au **tSampleRow**, où elles seront filtrées en fonction de leur position. Dans ce scénario, on suppose que le flux d'entrée contient les noms des vendeurs associés au nombre de produits qu'ils ont chacun vendus ainsi que le nombre d'années passées dans l'entreprise. Le résultat de l'opération de filtrage apparaît dans la console **Run**.

- A partir de la Palette, cliquez déposez les composants suivants dans l'éditeur graphique : le **tRowGenerator**, le **tSampleRow** et le **tLogRow**.
- Connectez ces trois composants à l'aide de liens de type **Row Main**.

- Dans l'éditeur graphique, sélectionnez le **tRowGenerator**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tRowGenerator**.
- Dans la vue **Basic settings**, définissez l'option **Schema Type** en mode **Built-In** puis cliquez sur le bouton **[...]** à côté du champ **Edit Schema** pour définir les données que vous voulez utiliser en entrée. Dans ce scénario, le schéma est composé de cinq colonnes.

- Dans l'onglet **Basic settings**, cliquez sur **RowGenerator Editor** pour définir les données à générer.
- Dans l'éditeur **RowGenerator Editor**, précisez le nombre de lignes à générer dans le champ **Number of Rows for RowGenerator** puis cliquez sur **OK** pour fermer l'éditeur.

- Dans l'éditeur graphique, sélectionnez le **tSampleRow**.
- Cliquez sur la vue **Component** pour définir la configuration de base (**Basic settings**) du **tSampleRow**.

- Dans l'onglet **Basic settings**, définissez l'option **Schema Type** en mode **Built-In** puis cliquez sur **Sync columns** pour récupérer le schéma à partir du composant **tRowGenerator**.
- Dans le panneau **Range**, définissez le filtre de sélection des lignes en utilisant la syntaxe appropriée comme indiqué plus haut. Dans ce scénario, on veut sélectionner la première et la cinquième ligne, ainsi que le groupe de lignes entre 9 et12.
- Dans l'espace graphique, sélectionnez le **tLogRow** puis cliquez sur la vue **Component** pour en définir la configuration de base (**Basic settings**). Pour plus d'informations sur les propriétés du **tLogRow**, consultez *tLogRow, page 1163*.
- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

```
Starting job sample_Row at 11:16 20/08/2008.
.--+-------------------+----------+---------+----------.
|                      tLogRow_1                        |
|=-+-------------------+----------+---------+----------=|
|ID|Years_in_Enterprise|First_Name|Last_Name|Sale_Values|
|=-+-------------------+----------+---------+----------=|
|1 |1                  |Martin    |Taft     |16         |
|5 |1                  |Franklin  |Adams    |45         |
|9 |1                  |William   |Madison  |15         |
|10|1                  |George    |Kennedy  |47         |
|11|1                  |George    |Van Buren|16         |
|12|1                  |Theodore  |Grant    |68         |
'--+-------------------+----------+---------+----------'

Job sample_Row ended at 11:16 20/08/2008. [exit code=0]
```

Les résultats filtrés apparaissent dans la console de log. Ils affichent la première et la cinquième ligne, ainsi que les lignes entre 9 et 12.

tSortRow

Propriétés du tSortRow

Famille de composant	Processing	
Fonction	Ce composant trie les données d'entrée basées sur une ou plusieurs colonnes, selon un type de tri et un ordre.	
Objectif	Il permet d'établir des métriques et des tables de classification.	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma dans cette vue, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent dans le Job.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Criteria	Cliquez sur **[+]** pour ajouter autant de lignes que nécessaire pour mettre en place le tri. La première colonne définie dans le schéma est sélectionnée par défaut.
		Schema column : Sélectionnez la colonne de votre schéma sur laquelle vous souhaitez baser votre tri. Notez que l'ordre est important car il détermine la priorité de tri.
		Sort type : Numérique ou Alphabétique. D'autres types de tri sont à venir.
		Order : Ordre ascendant ou descendant.
Advanced settings	*Sort on disk*	Personnalisez la mémoire utilisée de manière temporaire pour stocker les données de sortie. **Temp data directory path** : Définissez le chemin d'accès au dossier dans lequel stocker les fichiers temporaires. **Create temp data directory if not exists** : Cochez cette case afin de créer le dossier s'il n'existe pas. **Buffer size of external sort** : Entrez la taille de la mémoire physique à allouer au processus de tri.

	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant manipule le flux de données et par conséquent requiert un composant en entrée et un composant en sortie. Il s'agit donc d'un composant intermédiaire.	
Limitation	n/a	

Scénario : Trier des données

Le scénario suivant décrit un Job constitué de trois composants. Un composant **tRowGenerator** est utilisé pour créer des entrées de façon aléatoire. Ces entrées seront ensuite envoyées au composant **tSortRow** afin d'être triées selon une valeur définie. Dans ce scénario, le flux d'entrée contient des noms de vendeurs ainsi que leur volume de vente respectif et leur nombre d'années d'ancienneté dans l'entreprise. Le résultat de l'opération de tri est affiché dans la console **Run**.

- Cliquez et déposez les trois composants requis pour ce scénario : **tRowGenerator**, **tSortRow** et **tLogRow**.
- Connectez-les à l'aide de connexions de type **Row Main**.
- Dans le **RowGenerator editor**, définissez les valeurs qui seront créées de manière aléatoire et qui seront ensuite triées par le **tSortRow**. Pour plus d'informations concernant l'utilisation du **tRowGenerator**, consultez *Propriétés du tRowGenerator, page 1204.*

- Dans ce scénario, chaque vendeur est classé en fonction de la valeur de ses ventes (*Sales*) et de son ancienneté dans l'entreprise (*YearsInComp*).
- Double-cliquez sur **tSortRow** pour afficher l'onglet **Basic settings**. Définissez la priorité de tri sur la valeur des ventes et, en second critère, sur l'ancienneté.

- Utilisez le bouton (+) pour ajouter le nombre de lignes de critères requis. Paramétrez le type de tri, dans cet exemple, les deux critères sont de type numérique. Enfin, étant donné que la sortie est une classification, définissez l'ordre de tri comme descendant.
- Dans l'onglet **Advanced Settings**, cochez l'option **Sort on disk** pour modifier les paramètres de la mémoire temporaire. Dans le champ **Temp data directory path**, renseignez le chemin d'accès au dossier dans lequel vous voulez stocker les données temporaires. Dans le champ **Buffer size of external sort**, définissez la taille maximale de la mémoire tampon à allouer au traitement des données.

La valeur par défaut de la mémoire tampon est de 1000000 mais plus vous traitez un nombre important de lignes et/ou colonnes, plus cette valeur devra être élevée pour éviter l'interruption automatique du Job signifiée par le message d'erreur "out of memory".

- Assurez-vous que ce flux est connecté au composant de sortie **tLogRow**, afin d'afficher le résultat dans la console **Run**.
- Appuyez sur **F6** pour exécuter le Job ou passez à la vue **Run** et cliquez sur le bouton **Run**. Le classement est d'abord basé sur la valeur des ventes puis sur le nombre d'années d'ancienneté.

Execution
Debug
Run
Kill
Clear before run
Watch
Stats & Traces
Statistics
Traces
Clear
Starting job SimpleSort at 15:22 02/01/2007.
5|2|Steffie|89
1|4|Mickael|75
6|1|Bertrand|60
4|3|Pierrick|39
2|4|Matthew|25
3|4|Fabrice|8
Job SimpleSort ended at 15:22 02/01/2007. [exit code=0]

Composants System

Ce chapitre passe en revue les principaux composants de la famille **System** de la **Palette** de **Talend Open Studio**.

La famille **System** regroupe les composants permettant d'interagir avec votre système d'exploitation.

tRunJob

Propriétés du tRunJob

Famille de composant	System	
Fonction	Le composant **tRunJob** exécute le Job spécifié dans ses propriétés dans un contexte défini.	
Objectif	Le **tRunJob** permet de maîtriser des systèmes de Job complexes devant être exécuter l'un après l'autre.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma dans cette vue, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent dans le Job.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	CopyChild Job Schema	Cliquez sur ce bouton pour récupérer le schéma du Job fils.
	Job	Sélectionnez le Job à appeler et à traiter. Assurez-vous que le Job appelé ait été exécuté au moins une fois auparavant pour que son exécution via le **tRunJob** se déroule le mieux possible.
	Version	Sélectionnez la version du Job fils que vous souhaitez utiliser.
	Context	Si vous avez paramétré des contextes et des variables pour les Jobs à exécuter via le **tRunJob**, sélectionnez dans la liste le contexte à appliquer.
	Use an independent process to run subjob	Cochez cette case afin d'utiliser un processus idépendant pour exécuter le sous-job. Cela vous permettra de résoudre des problèmes liésà la taille limite de la mémoire.

	Die on child error	Décochez cette case pour exécuter le Job parent même s'il y a une erreur lors de l'exécution du Job fils.
	Transmit whole context	Cochez cette case pour obtenir les contextes du Job parent. Décochez-la pour obtenir les contextes du Job fils.
	Context Param	Vous pouvez modifier les paramètres de contexte sélectionnés. Cliquez sur le bouton **(+)** pour ajouter les paramètres tels qu'ils ont été définis dans le contexte du Job fils. Pour plus d'informations sur les paramètres de contexte, consultez la section *Définir les paramètres de contexte* du **Guide Utilisateur** de **Talend Open Studio**.
Advanced settings	*Print Parameters*	Cochez cette case pour afficher les paramètres internes et externes dans la **Console**.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en Job standalone ou peut aider à clarifier certains Jobs complexes en évitant d'avoir un nombre important de sous-jobs dans le même Job.	
Global Variables		**Child return code** : indique le code de retour Java du Job fils. Disponible comme variable **After**. Retourne un entier : sans erreur > la valeur du code est 0. avec erreur(s) > un message d'erreur apparaît. **Child exception stack trace** : retourne la trace de la pile Java d'un Job fils. Disponible comme variable **After**. Retourne une chaîne de caractères. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur** **Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject, Iterate. **Trigger** : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.

Limitation	n/a

Scénario : Exécuter un Job fils

Le scénario suivant décrit un Job à un composant appelant et exécutant un autre Job. Le Job à exécuter lit une liste de fichiers délimités de base et affiche son contenu dans la console de la vue **Run**. La particularité de ce Job réside dans le fait que ce dernier est exécuté à partir d'un autre Job et utilise la variable de contexte provenant du fichier d'entrée devant être traité.

Créez le premier Job lisant un fichier délimité.

- A partir de la **Palette**, cliquez et déposez les composants **tFileInputDelimited** et **tLogRow** dans l'espace de modélisation.
- Connectez ces deux composants à l'aide d'un lien de type **Row Main**.
- Double-cliquez sur le composant **tFileInputDelimited** pour afficher la vue **Basic settings** et paramétrer les propriétés du composant d'entrée.
- Dans la liste déroulante **Property Type**, sélectionnez l'option **Built-In**.

- Cliquez dans le champ **File Name** puis appuyez sur la touche **F5** pour ouvrir la fenêtre de configuration de variable.
- Nommez cette nouvelle variable de contexte, appelez-la *File*.

- Pour cet exemple, laissez les paramètres par défaut, i.e. ne cochez pas la case **Prompt for value** et laissez le champ **Prompt** vide.

- Cliquez sur **Finish** pour valider puis appuyez sur **Enter** pour vous assurer que la nouvelle variable de contexte est conservée dans le champ **File Name**.
- Dans les champs **Field separator** et **Row separator** de l'onglet **Basic settings**, renseignez respectivement les séparateurs de champs et de lignes utilisés dans le fichier d'entrée.
- Dans cet exemple, les fichiers ne possèdent ni en-tête ni pied de page. Ainsi, laissez les champs **Header** et **Footer** vides. Les fichiers sont relativement courts, vous pouvez donc laisser le champ **Limit** vide.
- Dans la liste déroulante **Schema type**, sélectionnez l'option **Built-in**. Cliquez sur le bouton **[...]** pour configurer manuellement le schéma.
- Ajoutez deux colonnes et nommez-les comme les deux premières colonnes du fichier d'entrée. Dans cet exemple : *username* et *age*.

  Si vous avez stocké le schéma dans le **Repository**, sélectionnez la métadonnée correspondant à la structure de votre fichier d'entrée.

- Double-cliquez sur le composant **tLogRow** pour afficher la vue **Basic settings** et paramétrer les propriétés du composant de sortie.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant d'entrée puis paramétrez les autres options selon vos besoins.
- Enregistrez le Job fils puis appuyez sur la touche **F6** vous assurer qu'il s'exécute correctement.

Créez le deuxième Job jouant le rôle de Job parent.

- Cliquez et déposez les composants **tFileList** et **tRunJob** dans l'espace de modélisation.
- Connectez ces deux composants à l'aide d'un lien de type **Iterate**.
- Double-cliquez sur le composant **tFileList** pour afficher la vue **Basic settings** et paramétrer les propriétés du composant.

- Dans le champ **Directory**, renseignez le chemin d'accès au répertoire contenant les fichiers à traiter ou cliquez sur le bouton **[...]** pour parcourir vos dossiers jusqu'à ce répertoire. Dans cet exemple, il s'agit du répertoire *tRunJob* qui contient trois fichiers délimités.

- Dans la liste déroulante **FileList Type**, sélectionnez l'option **Files**.
- Cochez la case **Use Glob Expressions as Filemask** pour pouvoir utiliser des expressions régulières dans vos masques.
- Cliquez sur le bouton **[...]** pour ajouter un masque et renseignez le filtre à appliquer. Dans cet exemple, les fichiers à récupérer sont de type délimité, le filtre choisi est "**.csv*".
- Double-cliquez sur le composant **tRunJob** pour afficher la vue **Basic settings** et paramétrer les propriétés du composant.

- Cliquez sur le bouton **[...]** à côté du champ Job pour accéder à la boîte de dialogue **[Find a Job]**.

- Sélectionnez le Job fils à exécuter puis cliquez sur **OK** pour fermer la boîte de dialogue. Le nom du Job sélectionné apparaît dans le champ **Job** de la vue **Basic settings** du **tRunJob**.
- Cliquez sur le bouton **Copy Child Job Schema** pour récupérer le schéma défini dans le Job fils.

- Cliquez sur le bouton **[+]** pour ajouter une ligne de paramètres dans le champ **Context Param**.
- Cliquez dans la cellule **Values** puis appuyez sur **Ctrl+Espace** pour accéder à la liste des variables de contexte et sélectionnez le contexte *tFileList-1.CURRENT_FILEPATH*. La cellule **Values** est automatiquement renseignée par la variable de contexte correspondante : `((String)globalMap.get("tFileList-1.CURRENT_FILEPATH"))`. Pour plus d'informations sur les paramètres de contexte, consultez la section *Définir les paramètres de contexte* du **Guide Utilisateur** de **Talend Open Studio**.
- Enregistrez le Job maître puis appuyez sur la touche **F6** pour l'exécuter.

```
Starting job tRunJob_father at 12:17 06/10/2009.

[statistics] connecting to socket on port 4043
[statistics] connected
clevenes|25
elsabot|32
mlelandais|26
hmassy|31
abibota|24
ychen|27
plegall|28
smallet|34
scorreia|37
[statistics] disconnected
Job tRunJob_father ended at 12:17 06/10/2009. [exit code=0]
```

Le Job appelé lit les données contenues dans les fichiers d'entrée comme défini dans le schéma d'entrée et le résultat de ce Job est directement affiché dans la console **Run**.

Voir également :

- Le composant *tLoop, page 1223*,
- Le *Scénario 1 : Mettre des données en tampon (Java), page 1177* du composant **tBufferOutput**.

tSetEnv

Propriétés du tSetEnv

Famille du composant	System	
Fonction	Le composant **tSetEnv** ajoute des variables à l'environnement du système de manière temporaire, pendant l'exécution du Job.	
Objectif	Le **tSetEnv** permet de créer des variables et d'exécuter un Job script en communicant les informations des variables nouvellement créées entre les différents sous-jobs. Après l'exécution du Job, ces nouvelles variables sont supprimées.	
Basic settings	*Parameters*	**Cliquez sur le bouton [+] pour ajouter les variables nécessaires au Job.** **name** : Renseignez la syntaxe de la nouvelle variable. **value** : Entrez une valeur pour ce paramètre en fonction du contexte. **append** : Cochez cette case pour ajouter la nouvelle variable à la fin.
Utilisation	Le **tSetEnv** peut être utilisé comme composant de début ou composant intermédiaire.	
Limitation	n/a	

Scénario : Modifier la variable *Date* lors de l'exécution d'un Job

Le scénario suivant est un Job qui permet de lire une colonne dans une base de données Oracle, de récupérer la date courante à partir de cette colonne en utilisant une requête de base de données de créer une nouvelle variable grâce au **tSetEnv** pour modifier le format de la date puis d'afficher la date modifiée dans la console.

Pour modifier le format de la date en utilisant une nouvelle variable créée par le **tSetEnv** :

- A partir de la **Palette**, cliquez-déposez les composants **tSetEnv**, **tOracleInput** et **tLogRow** dans l'éditeur graphique.
- Connectez le **tSetEnv** au **tOracleInput** à l'aide d'un lien **OnSubjobOk**.
- Connectez le **tOracleInput** au **tLogRow** à l'aide d'un lien **Row Main**.

- Sélectionnez le **tSetEnv** dans l'éditeur graphique puis cliquez sur la vue **Component**.
- Dans l'onglet **Basic settings**, cliquez sur le bouton **[+]** pour ajouter une ligne de paramètres et définissez votre nouvelle variable.
- Cliquez dans la cellule **name** puis renseignez la syntaxe de la variable concernant la date. Dans cet exemple, on utilise `NLS_DATE_FORMAT`.
- Cliquez dans la cellule **value** puis renseignez la valeur désirée pour votre nouvelle variable de date.

Dans cet exemple, on veut transformer le format de date `DD-MMM-YY` prédéfini par le système en `YYYY-MM-DD`.

- Sélectionnez le **tOracleInput** puis cliquez sur l'onglet **Component** .
- Paramétrez la configuration de base (**Basic settings**) du composant **tOracleInput**. Pour plus d'informations, consultez le composant *tOracleInput, page 582*.

Dans cet exemple, on effectue une requête sur une base Oracle pour extraire les données contenues dans la colonne the “REGISTERDATE” de la table “NEW_CUSTOMERS_MLE”.

- Sélectionnez le **tLogRow** puis cliquez sur la vue **Component**.
- Renseignez la configuration de base (**Basic settings**) du composant **tLogRow**. Pour plus d’informations, consultez le composant *tLogRow, page 1163*.
- Enregistrez votre Job puis appuyez sur **F6** pour l’exécuter.

```
Starting job oracle at 10:30 24/07/2008.
.------------.
|  tLogRow_1 |
+------------+
| right_now  |
+------------+
| 2008-07-24 |
'------------'
===
execution time: 85 milliseconds
===
Job oracle ended at 10:30 24/07/2008. [exit code=0]
```

La date s’affiche dans la console au format `YYYY-MM-DD` modifié par le composant **tSetEnv.**

Pour afficher la date au format pré-défini par le système en utilisant ce Job :

- Dans l’espace graphique, cliquez-droit sur le composant **tSetEnv** et sélectionnez l’option **Deactivate tSetEnv_1** à partir de la liste déroulante.
- Enregistrez votre Job puis appuyez sur **F6** pour l’exécuter.

```
Starting job oracle at 10:31 24/07/2008.
.-----------.
| tLogRow_1 |
+-----------+
| right_now |
+-----------+
| 24-JUL-08 |
'-----------'
===
execution time: 95 milliseconds
===
Job oracle ended at 10:31 24/07/2008. [exit code=0]
```

La date s'affiche dans la console au format `DD-MMM-YY` pré-défini par le système.

tSSH

Propriétés du tSSH

Famille de composant	System	
Fonction	Le composant **tSSH** renvoie les données d'un ordinateur distant via le protocole Secure Shell défini.	
Objectif	Le **tSSH** permet d'entrer en communication avec un serveur distant et de renvoyer les informations de sécurité.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma dans cette vue, il devient automatiquement Built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent dans le Job.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Host	Adresse IP.
	Port	Numéro du port d'écoute.
	User	Informations d'authentification de l'utilisateur.
Authentication method	*Public Key*/Key *Passphrase*/*Private Key*	Sélectionnez l'option adéquate. Si vous choisissez l'option **Public Key**, saisissez la passphrase (phrase secrète), si nécessaire, dans le champ **Key passphrase**. Puis, dans le champ **Private Key**, saisissez la clé privée ou cliquez sur le bouton **[...]** à côté du champ **Private Key** pour parcourir votre répertoire jusqu'à la clé privée.
Authentication method	*Password*/*Password*	Sélectionnez l'option adéquate. Si vous choisissez l'option **Password**, saisissez votre mot de passe dans le champ **Password**.
Authentication method	*Keyboard Interactive*/*Password*	Sélectionnez l'option adéquate. Si vous choisissez l'option **Keyboard Interactive**, saisissez le mot de passe requis dans le champ **Password**.

	Pseudo terminal	Cochez cette case pour appeler un interpréteur de commandes interactif effectuant les opérations de terminal.
	Command separator	Saisissez le séparateur de commandes requis. Lorsque la case **Pseudo terminal** est cochée, ce champ n'est plus disponible.
	Commands	Saisissez la commande permettant de récupérer l'information de l'ordinateur distant. Lorsque la case **Pseudo terminal** est cochée, ce tableau devient un émulateur de terminal, et chaque ligne du tableau devient une commande.
	Use timeout/timeout in seconds	Définit la durée avant expiration. Un message notifiant l'expiration sera généré si l'actuel temps de réponse dépasse ce délai d'expiration.
	Standard Output	Sélectionnez la destination de la sortie standard à retourner La sortie peut être retournée vers : - la console (**to console**) : la sortie s'affiche dans la console de la vue **Run**. - les variables globales (**to global variables**) : la sortie est indiquée par la variable globale correspondante. - la console et les variables globales (**both to console and global variable**) : la sortie est indiquée par les deux moyens. - une sortie normale (**normal**) : la sortie est une sortie SSH standard.
	Error Output	Sélectionnez la destination de la sortie d'erreurs à retourner. La sortie peut être retournée vers : - la console (**to console**) : la sortie s'affiche dans la console de la vue **Run**. - les variables globales (**to global variables**) : la sortie est indiquée par la variable globale correspondante. - la console et les variables globales (**both to console and global variable**) : la sortie est indiquée par les deux moyens. - une sortie normale (**normal**) : la sortie est une sortie SSH standard.
Utilisation	Ce composant peut être utilisé en standalone.	

Global variables		**Standard Output** : Indique la sortie standard de l'exécution de la commande distante. Disponible en tant que variable **After**. Retourne une chaîne de caractères. **Error output** : Indique la sortie d'erreurs de l'exécution de la commande distante. Disponible en tant que variable **After**. Retourne une chaîne de caractères. **Exit value** : Indique le statut de sortie de la commande distante. Disponible en tant que variable **After**. Retourne un entier. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur Talend Open Studio**.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main. **Trigger** : Run if, On Component Ok, On Component Error, On Subjob Ok, On Subjob Error. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Iterate. **Trigger** : Run if, On Component Ok, On Component Error, On Subjob Ok, On Subjob Error. Pour plus d'informations concernant les connexions, consultez *Types de connexions* du **Guide Utilisateur Talend Open Studio**.
Limitation	L'utilisation de ce composant est adaptée pour les systèmes de type Unix.	

Scénario : Accéder aux informations du système distant via SSH

Le scénario suivant décrit un Job utilisant un protocole SSH pour afficher le nom d'hôte du serveur distant auquel vous souhaitez vous connecter et la date actuelle de ce système distant.

Un composant **tSSH** est suffisant pour ce Job. Cliquez et déposez-le de la famille **System** de la **Palette**.

Double-cliquez sur le **tSSH** et sélectionnez l'onglet **Basic settings** de la vue **Component**.

- Renseignez le nom de l'hôte dans le champ **Host** auquel accéder à partir du SSH, ainsi que le numéro du port dans le champ **Port**.
- Dans le champ **User**, renseignez le nom d'identification de l'utilisateur sur l'ordinateur distant.
- Sélectionnez la méthode d'authentification dans la liste déroulante **Authentication method**. Dans cet exemple, la méthode d'authentification utilisée est **Public key** (clé publique).
- Renseignez donc le champ **Private key** avec votre clé privée.
- Dans le champ **Command**, saisissez la commande suivante. Dans cet exemple, saisissez `hostname; date` entre guillemets simples (puisque le Job est généré en Perl).
- Cochez la case **Use timeout** et paramétrez la durée avant expiration à 5 secondes.

```
Starting job uniteElisa at 16:26 26/09/2007.
picasso
Wed Sep 26 14:24:15 CEST 2007
Job uniteElisa ended at 16:26 26/09/2007. [exit code=0]
```

L'ordinateur distant renvoie le nom de l'hôte, ainsi que la date et l'heure en cours.

tSystem

Propriétés du tSystem

Famille de composant	System	
Fonction	Le composant **tSystem** exécute une ou plusieurs commande(s) système.	
Objectif	**tSystem** peut appeler d'autres commandes de traitement, fonctionnant déjà dans un Job plus grand.	
Basic settings	*Use home directory*	Cochez cette case pour changer le nom et le chemin du répertoire dédié.
	Command	Saisissez la commande système à exécuter. Notez que la syntaxe n'est pas vérifiée. *Sous Windows, les commandes MS-DOS ne permettent pas de passer directement du dossier courant au dossier contenant le fichier à exécuter. Pour exécuter un fichier, vous devez donc utiliser une première commande afin de changer de dossier courant, puis une deuxième commande exécutant le fichier.*
	Standard Output et *Error Output*	Sélectionnez le type de sortie auquel les données traitées seront transférées.
		to console : transmet les données vers la console de la vue **Run**.
		to global variable : les données sont envoyées dans une variable de sortie liée au composant **tSystem**.
		both to console and to global variable : les données sont envoyées à la fois vers la console de la vue **Run** et dans une variable de sortie liée au composant **tSystem**.
		normal : les données sont envoyées au composant suivant.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma dans cette vue, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent dans le Job.

		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de Talend Open Studio.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de Talend Open Studio.
	Environment variables	Cliquez sur le bouton **[+]** pour ajouter les variables d'environnement nécessaires au Job. **name** : Renseignez la syntaxe de la nouvelle variable. **value** : Entrez une valeur pour ce paramètre en fonction du contexte.
Utilisation	Ce composant sert aux entreprises qui utilisent déjà d'autres applications qu'ils souhaitent intégrer à leur flux via Talend.	
Global Variables		**Standard Output** : retourne la sortie standard d'un processus. Disponible comme variable **After**. Retourne une chaîne de caractères. **Error Output** : retourne la sortie erronée d'un processus. Disponible comme variable **After**. Retourne une chaîne de caractères. **Exit Value** : retourne un code de sortie. Disponible comme variable **After**. Retourne un entier : -sans d'erreur > le code de sortie est 0. -avec erreurs(s) > le code de sortie est 1. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* dans le **Guide Utilisateur** Talend Open Studio.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main. **Trigger** : OnSubjobOk, OnSubjobError, Run if. Liens d'entrée (d'un composant à un autre) : **Row** : Main, Reject, Iterate. Trigger : OnSubjobOk, OnSubjobError, Run if, OnComponentOk, OnComponentError, Synchronize, Parallelize. Pour plus d'informations concernant les liens, consultez *Types de connexions* dans le **Guide Utilisateur** Talend Open Studio.
Limitation	n/a	

Scénario : Echo 'Hello World!'

Ce scénario est constitué d'un Job à deux composants permettant d'afficher un message dans la console de la vue **Run**.

- A partir de la **Palette**, déposez les composants **tSystem** et **tJava** dans l'espace graphique.
- Cliquez-droit sur le composant **tSystem** et sélectionnez un lien de type **OnSubjobOk** pour relier ces deux composants. Lorsque vous exécutez le Job, le premier composant déclenchera le deuxième.
- Double-cliquez sur le composant **tSystem** pour afficher l'onglet **Basic settings** et configurer ses paramètres de base.

- Dans le champ **Command**, saisissez la commande `echo` suivie de la chaîne de caractères à afficher : `"cmd /c echo HELLO WORLD"`
- Dans le champ **Standard Output**, sélectionnez l'option **to a global variable** pour envoyer la sortie dans une variable globale.
- Laissez les autres champs par défaut.
- Sélectionnez ensuite le composant **tJava**.

- Saisissez la commande Java permettant d'afficher la variable de sortie du **tSystem** dans la console : `System.out.println("Hello World!");`
- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter.

```
Starting job tSystem_scenario at 17:26 08/10/2009.

[statistics] connecting to socket on port 3961
[statistics] connected
Hello World!
[statistics] disconnected
Job tSystem_scenario ended at 17:26 08/10/2009. [exit code=0]
```

Le Job exécute une commande `echo` et affiche la sortie dans la console de log de la vue **Run** grâce à la fonction `println` de la commande Java du composant **tJava**.

Composants Talend MDM

Ce chapitre passe en revue les propriétés des principaux composants présents dans la famille **Talend MDM** de la **Palette** de **Talend Open Studio**.

La famille **Talend MDM** regroupe les composants qui lisent et écrivent des données dans les fichiers XML du hub XML MDM.

tMDMBulkLoad

Propriétés du tMDMBulkLoad

Famille de composant	Talend MDM	
Fonction	Le composant **tMDMBulkLoad** écrit en masse des données maître structurées XML dans le hub MDM.	
Objectif	Ce composant utilise le mode bulk pour écrire des données, de façon à ce que de gros lots de données ou des données très complexes puissent être chargé(e)s rapidement sur le serveur MDM.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Si vous modifiez le schéma, il redevient automatiquement built-in. Cliquez sur le bouton **Sync columns** afin de récupérer le schéma du composant précédent.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	XML Field	Sélectionnez le nom de la colonne dans laquelle vous souhaitez écrire les données XML.
	URL	Saisissez l'URL d'accès au serveur MDM.
	Username et *Password*	Informations d'authentification de l'utilisateur du serveur MDM.
	Version	Saisissez le nom de la version des données maître à laquelle vous souhaitez vous connecter, en fonction de vos droits utilisateur. Laissez ce champ vide pour afficher la version par défaut des données maître.
	Data model	Saisissez le nom du modèle de données par rapport auquel vous souhaitez valider les données à écrire.
	Data Container	Saisissez le nom du conteneur de données contenant les données que vous souhaitez écrire.
	Entity	Saisissez le nom de l'entité contenant les données que vous souhaitez effacer.

	Validate	Cochez cette case afin de valider les données que vous souhaitez écrire sur le serveur MDM, par rapport à des règles de validation définies pour le modèle de données courant. Pour plus d'informations concernant la définition des règles de validation, consultez le Guide **Talend Master Data Management Administrator Guide** (en anglais). *Si vous préférez un chargement plus rapide, ne cochez pas cette case.*
	Generate ID	Cochez cette case pour générer un identifiant aux données écrites. *Si vous préférez un chargement plus rapide, ne cochez pas cette case.*
	Commit size	Saisissez le nombre de lignes de chaque lot à écrire sur le serveur MDM.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Connections		Liens de sortie (d'un composant à un autre) : **Row** : Main. **Trigger** : Run If, OnComponentOk, OnComponentError, OnSubjobOk, OnSubjobError. Liens d'entrée (d'un composant à un autre) : **Row** : Main. **Trigger** : Run If, OnComponentOk, OnComponentError, OnSubjobOk, OnSubjobError. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur Talend Open Studio**.
Utilisation	Ce composant nécessite toujours un flux d'entrée afin de stucturer des données en XML. Si vos données ne sont pas encore structurées XML, vous devez utiliser des composants tels que le **tWriteXMLField** afin de transformer ces données en XML. Pour plus d'informations concernant le **tWriteXMLField**, consultez *tWriteXMLField, page 1472*.	

Scénario : Charger des enregistrements dans l'entité métier ProductFamily

Ce scénario décrit un Job qui charge des enregistrements dans l'entité métier *ProductFamily* définie par un modèle de données spécifique dans le hub MDM.

Prérequis pour ce Job :

- Le conteneur de données *Product* : ce conteneur de données est utilisé pour séparer le domaine de données maître des produits des autres domaines de données maître.
- Le modèle de données *Product* : ce modèle de données est utilisé pour définir les attributs, les règles de validation, les droits d'accès des utilisateurs et les relations des entités en rapport. Il définit les attributs de l'entité métier *ProductFamily*.

- L'entité métier *ProductFamily* : cette entité métier contient les *Id* et *Name*, définis dans le modèle de données *Product*.

Pour plus d'informations concernant la création d'un conteneur de données, un modèle de données et une entité métier ainsi que ses attributs, consultez le **Guide Talend Master Data Management Administrator Guide** (en anglais).

Le Job dans ce scénario utilise trois composants.

- Un **tFixedFlowInput** : ce composant génère les enregistrements à charger dans l'entité métier *ProductFamily*. Généralement, les enregistrements à charger sont volumineux et stockés dans un fichier spécifique, cependant, pour faciliter la reproduction de ce scénario, ce Job utilise un **tFixedFlowInput** afin de générer quatre exemples d'enregistrements.
- Un **tWriteXMLField** : ce composant transforme les données d'entrée pour leur donner une structure XML.
- Un **tMDMBulkLoad** : ce composant écrit en masse les données d'entrée dans l'entité métier *ProductFamily* et génère une valeur d'ID pour chaque donnée de l'enregistrement.

Pour reproduire ce scénario, procédez comme suit :

- Déposez un **tFixedFlowInput**, un **tWriteXMLField** et un **tMDMBulkLoad** de la **Palette** dans l'espace de modélisation graphique.
- Cliquez-droit sur le **tFixedFlowInput** afin d'ouvrir son menu contextuel.
- Sélectionnez **Row > Main** pour relier le **tFixedFlowInput** au composant suivant à l'aide d'un lien **Main**.
- Répétez l'opération pour lier les autres composants.
- Double-cliquez sur le **tFixedFlowInput** afin d'ouvrir sa vue **Basic settings**.

- Cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour ouvrir l'éditeur du schéma.

- Dans l'éditeur du schéma, cliquez sur le bouton **[+]** pour ajouter une ligne.
- Cliquez sur la nouvelle et saisissez son nom : *family*.
- Cliquez sur **OK**.
- Dans la zone **Mode** de la vue **Basic settings**, sélectionnez l'option **Use inline table**.
- Sous le tableau, cliquez sur le bouton **[+]** quatre fois afin d'ajouter quatre lignes au tableau.
- Dans le tableau, cliquez sur chacune des lignes ajoutées et saisissez leur nom, entre guillemets : *Shirts*, *Hats*, *Pets*, *Mugs* (Chemises, chapeaux, vêtements pour animaux, tasses).
- Double-cliquez sur le composant **tWriteXMLField** afin d'ouvrir sa vue **Basic settings**.

- Cliquez sur le bouton **[...]** à côté du champ **Edit schema** afin d'ouvrir l'éditeur du schéma, dans lequel vous pouvez ajouter une ligne en cliquant sur le bouton **[+]**.

- Cliquez sur la nouvelle ligne, dans la partie droite de l'éditeur du schéma et saisissez le nom de la colonne de sortie dans laquelle vous souhaitez écrire le contenu XML. Dans cet exemple, saisissez *xmlRecord*.
- Cliquez sur **OK** pour valider le schéma de sortie et fermer l'éditeur du schéma.
- Dans la boîte de dialogue qui s'ouvre, cliquez sur **OK** afin de propager le schéma au composant suivant.
- Dans la vue **Basic settings**, cliquez sur le bouton **[...]** à côté du champ **Configure Xml Tree** pour ouvrir l'interface permettant de créer la structure XML.

- Dans la zone **Link Target**, cliquez sur *rootTag* et renommez le champ *ProductFamily*, qui est le nom de l'entité métier utilisée dans ce scénario.
- De la zone **Linker source**, glissez-déposez *family* dans *ProductFamily*, dans la zone **Link target**.
  Une boîte de dialogue s'affiche et vous demande quel type d'opération vous souhaitez effectuer.
- Sélectionnez **Create as sub-element of target node** afin de créer un sous-élément du noeud *ProductFamily*. L'élément *family* apparaît sous le noeud *ProductFamily*.
- Dans la zone **Link target**, cliquez sur le noeud *family* et renommez-le *Name*, qui est l'un des attributs de l'entité métier *ProductFamily*.
- Cliquez-droit sur le noeud *Name*et sélectionnez **Set As Loop Element** dans le menu contextuel.
- Cliquez sur **OK** afin de valider la structure XML que vous avez définie.
- Double-cliquez sur le **tMDMBulkLoad** afin d'ouvrir sa vue **Basic settings**.

- Dans la liste **XML Field**, sélectionnez *xmlRecord*.
- Dans le champ **URL**, saisissez l'URL du serveur MDM, entre guillemets : par exemple, *http://localhost:8080/talend/TalendPort*.
- Dans les champs **Username** et **Password**, saisissez respectivement votre identifiant et votre mot de passe de connexion au serveur MDM.
- Dans les champs **Data Model** et **Data Container**, saisissez les noms correspondants au modèle de données et au conteneur de données que vous souhaitez utiliser, *Product* pour les deux champs, dans ce scénario.
- Dans le champ **Entity**, saisissez le nom de l'entité métier dont les enregistrements doivent être chargés. Dans cet exemple, saisissez *ProductFamily*.
- Cochez la case **Generate ID** afin de générer des valeurs d'ID pour les enregistrements à charger.
- Appuyez sur **F6** pour exécuter le Job.

- Connectez-vous à l'interface Web de **Talend Master Data Management** pour vérifier les nouveaux enregistrements de l'entité métier *ProductFamily*.

tMDMDelete

Propriétés du tMDMDelete

Famille de composant	Talend MDM	
Fonction	Le composant **tMDMDelete** supprime des enregistrements de données d'entités spécifiques du hub MDM (référentiel XML).	
Objectif	Ce composant efface des données maître du hub MDM.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Si vous modifiez le schéma, il redevient automatiquement built-in. Cliquez sur le bouton **Sync columns** afin de récupérer le schéma du composant précédent.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	URL	Saisissez l'URL d'accès au serveur MDM.
	Username et *Password*	Informations d'authentification de l'utilisateur du serveur MDM.
	Version	Saisissez le nom de la version des données maître à laquelle vous souhaitez vous connecter, en fonction de vos droits utilisateur. Laissez ce champ vide pour afficher la version par défaut des données maître.
	Entity	Saisissez le nom de l'entité contenant les données que vous souhaitez effacer.
	Data Container	Saisissez le nom du conteneur de données contenant les données que vous souhaitez effacer.
	Keys	Spécifiez le(s) champ(s) (dans l'ordre de la séquence) composant la clé lorsque l'entité possède une clé multiple.

	Logical delete	Cochez cette case afin d'envoyer les données maître dans la Corbeille, et renseignez le chemin d'accès à la Corbeille (**Recycle bin path**). Une fois dans la Corbeille, les données maître peuvent être supprimées définitivement ou restaurées. Si vous laissez la case décochée, les données maître seront supprimées définitivement.
	Die on error	Cochez cette case afin d'arrêter le Job en cas d'erreur.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Utilisez ce composant pour écrire un fichier et séparer les champs à l'aide d'un séparateur spécifique.	

Scénario : Supprimer des données maître d'un hub MDM

Ce scénario décrit un Job à deux composants supprimant les enregistrements de données spécifiés du référentiel XML MDM.

- Glissez-déposez de la **Palette** dans l'espace de modélisation graphique un composant **tMDMInput** et un composant **tMDMDelete**, de la famille **Talend MDM**.
- Connectez les deux composants à l'aide d'un lien **Row** > **Main**.
- Double-cliquez sur le composant **tMDMInput** afin d'afficher sa vue **Component** et définir ses propriétés.

- Dans la liste **Property Type**, sélectionnez **Built-in** pour renseigner les champs manuellement. Si vous avez stocké votre connexion MDM dans les métadonnées du Repository, sélectionnez **Repository** dans la liste, et les champs seront automatiquement remplis.
- Dans la liste **Schema Type**, sélectionnez **Built-in** et cliquez sur le bouton **[...]** à côté du champ **Edit schema**, afin d'ouvrir une boîte de dialogue dans laquelle vous pourrez dfinir la structure des données maître que vous souhaitez lire dans le hub MDM.

- Les données maître sont collectées dans un schéma composé de quatre colonnes de type **String**: *Id*, *Name*, *City* et *State*. Cliquez sur **OK** pour fermer la boîte de dialogue et procéder à l'étape suivante.
- Dans le champ **URL**, saisissez entre guillemets l'URL du serveur MDM : *"http://localhost:8080/talend/TalendPort"*.
- Dans les champs **Username** et **Password**, saisissez votre identifiant et votre mot de passe de connexion au serveur MDM.
- Dans le champ **Version**, saisissez entre guillemets le nom de la version des données maître à laquelle vous souhaitez accéder. Laissez ce champ vide afin d'afficher la version des données maître par défaut.
- Dans le champ **Entity**, saisissez entre guillemets le nom de l'entité métier contenant les enregistrements de données que vous souhaitez lire. Ici, travaillez avec l'entité *Agency*.
- Dans le champ **Data Container**, saisissez entre guillemets le nom du conteneur de données contenant les données maître que vous voulez lire. Ici, utilisez le conteneur *DStar*.

La case **Use multiple conditions** est cochée par défaut.

•Dans le tableau **Operations**, définissez les conditions de filtre des données maître que vous souhaitez supprimer :
-cliquez sur le bouton **[+]** afin d'ajouter une nouvelle ligne.
Dans la colonne **Xpath**, saisissez entre guillemets le chemin Xpath et la balise du noeud XML sur lequel vous souhaitez appliquer le filtre. Ici, travaillez avec l'entité **Agency**, saisissez *"Agency/Id"*.
Dans la colonne **Function**, sélectionnez la fonction que vous souhaitez utiliser. Dans ce scénario, utilisez la fonction *Starts With*.
Dans la colonne **Value**, saisissez la valeur de votre filtre. Ici, l'objectif est de filtrer les données maître dont l'ID commence par *"TA"*.

- Dans la vue **Component**, cliquez sur l'onglet **Advanced settings** afin de configurer les paramètres avancés.

- Dans le champ **Loop XPath query**, saisissez entre guillemets la structure et le nom du noeud XML sur lequel exécuter la boucle.
- Dans la zone **Mapping** et dans la colonne **XPath query**, saisissez entre guillemets le nom de la balise XML sur laquelle vous souhaitez collecter les données maître, à côté de la colonne de sortie correspondante.
- Dans l'espace de modélisation graphique, cliquez sur le composant **tMDMDelete** afin d'afficher sa vue **Component** et définir ses propriétés.

- Dans la liste **Schema**, sélectionnez **Built-in** et cliquez sur le bouton **[...]** à côté du champ **Edit Schema**, afin de décrire la structure des données maître du hub MDM.

- Cliquez sur le bouton **[+]**, à droite, afin d'ajouter une colonne de type **String**. Nommez-la *xmlOutput*. Cliquez sur **OK**.
- Dans le champ **URL**, saisissez l'URL requise pour vous connecter au serveur MDM, par exemple : *"http://localhost:8080/talend/TalendPort"*.
- Dans les champs **Username** et **Password**, saisissez les informations d'authentification requises afin de vous connecter au serveur.
- Dans le champ **Version**, saisissez entre guillemets le nom de la version des données maître à laquelle vous souhaitez accéder. Laissez le champ vide si vous souhaitez accéder à la version par défaut.
- Dans le champ **Entity**, saisissez le nom de l'entité métier contenant les données maître que vous souhaitez supprimer. Ici, travaillez avec l'entité *Agency*.
- Dans le champ **Data Container**, saisissez le nom du conteneur de données contenant les donnéess à supprimer. Ici, utilisez le conteneur *DStar*.
- Dans le tableau **Keys**, cliquez sur le bouton **[+]** afin d'ajouter une nouvelle ligne. Dans la colonne **Keys**, sélectionnez la colonne contenant la clé de l'entité *Agency*. Ici, la clé de l'entité *Agency* est dans le champ *Id*.

  Si l'entité possède des clés multiples, ajoutez autant de lignes que nécessaire pour les clés, et sélectionnez-les dans l'ordre séquentiel.

- Cochez la case **Logical delete** si vous ne voulez pas supprimer définitivement les données maître. Elles seront envoyées dans la **Corbeille**. Une fois dans la **Corbeille**, les données maître peuvent être supprimées définitivement, ou bien restaurées. Si vous laissez cette case décochée, les données maître seront supprimées définitivement.
- Remplissez le champ **Recycle bin path**. Ici, vous avez laissé le chemin d'accès à la Corbeille par défaut, mais si votre chemin d'accès à la Corbeille est différent de celui par défaut, spécifiez-le.
- Appuyez sur **Ctrl+S** pour sauvegarder votre Job, puis sur **F6** pour l'exécuter.

Les données dont l'ID commence par "TA" ont été supprimées et envoyées dans la Corbeille MDM.

tMDMInput

Propriétés du tMDMInput

Famille de composant	Talend MDM	
Fonction	Le composant **tMDMInput** lit les données maître du hub MDM (référentiel XML).	
Objectif	Ce composant lit les données maître dans un hub MDM et rend possible le traitement de ces données.	
Basic settings	*Property Type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Si vous modifiez le schéma, il redevient automatiquement built-in.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	URL	Saisissez l'URL d'accès au serveur MDM.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur du serveur MDM.
	Version	Saisissez le nom de la version des données maître à laquelle vous souhaitez vous connecter, en fonction de vos droits utilisateur. Laissez ce champ vide pour afficher la version par défaut des données maître.
	Entity	Saisissez le nom de l'entité contenant les données que vous souhaitez effacer.
	Data Container	Saisissez le nom du conteneur de données contenant les données que vous souhaitez effacer.

	Use multiple conditions	Cochez cette case pour filtrer les données maître à l'aide de conditions. **Xpath** : saisissez, entre guillemets, le noeud XML sur lequel appliquer la condition. **Function** : sélectionnez la condition à utiliser. **Value** : saisissez, entre guillemets, la valeur souhaitée. **Predicate** : sélectionnez un prédicat lorsque vous utilisez plusieurs conditions. Si vous décochez cette case, vous avez la possibilité de sélectionner des ID particuliers devant être affichés dans la colonne **ID value** du tableau **IDS**. Si vous décochez la case **Use multiple conditions**, l'option **Batch Size** de l'onglet **Advanced settings** ne sera plus disponible.
	Skip rows	Renseignez le nombre de lignes à ignorer.
	Limit	Nombre maximum de lignes à traiter. Si Limit = 0, aucune ligne n'est lue ou traitée.
Advanced settings	*Batch Size*	Nombre de lignes dans chaque lot traité. Cette option n'apparaît pas si vous avez décoché la case **Use multiple conditions** dans l'onglet **Basic settings**.
	Loop XPath query	Noeud de l'arborescence XML sur lequel est basé la boucle.
	Mapping	**Column** : reflète le schéma tel qu'il est défini dans l'éditeur Edit Schema. **XPath query** : Saisissez les champs à extraire de la structure XML d'entrée. **Get Nodes** : Cochez cette case pour récupérer les balises XML entournant les données avec celles-ci.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est utilisé comme composant de début. Il nécessite un flux de sortie..	

Scénario : Lire les données maître d'un hub MDM

Ce scénario décrit un Job à deux composants qui lit les données maître dans un serveur MDM et les affiche dans la console de log.

- A partir de la **Palette**, glissez les composants **tMDMInput** de la famille **Talend MDM**, et **tLogRow** de la famille **Logs & Errors** dans l'espace de modélisation.
- Reliez-les via un lien de type **Row Main**.

- Double-cliquez sur le composant **tMDMInput** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés.

- Dans la liste **Property Type**, sélectionnez **Built-In** et renseignez les champs manuellement. Si vous avez stocké les informations de connexion au MDM dans les métadonnées du Repository, sélectionnez **Repository** dans la liste et les champs suivants seront renseignés automatiquement.
- Dans la liste **Schema**, sélectionnez **Built-In** et cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour décrire la structure des données maître que vous souhaitez lire sur le serveur MDM.

- Les données maître sont récupérées dans un schéma à trois colonnes de type **String** : *ISO2Code*, *Name*, *Currency*. Cliquez sur **OK**.
- Dans le champ **URL**, saisissez entre guillemets l'URL de connexion au serveur MDM.

- Dans les champs **Username** et **Password**, saisissez l'identifiant et le mot de passe de connexion au serveur MDM.
- Dans le champ **Version**, saisissez entre guillemets le nom de la version des données maître à laquelle vous avez souhaitez accéder. Laissez le champ vide pour afficher la version par défaut.
- Dans le champ **Entity**, saisissez entre guillemets le nom de l'entité métier contenant les données maître que vous souhaitez lire.
- Dans le champ **Data Container**, saisissez entre guillemets le nom du conteneur de données contenant les données maître que vous souhaitez lire.
- Dans la vue **Component**, cliquez sur l'onglet **Advanced settings** pour paramétrer les propriétés avancées du composant.

- Dans le champ **Loop XPath query**, saisissez entre guillemets l'arborescence et le nom du noeud XML sur lequel effectuer la boucle.
- Dans le tableau **Mapping**, les colonnes du schéma du composant apparaissent sous la colonne Column. Saisissez entre guillemets le nom de la balise XML dans laquelle récupérer les données maître dans le champ **XPath query** correspondant à la colonne du schéma de sortie.
- Dans l'espace de modélisation, double-cliquez sur le composant **tLogRow** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés.
- Cliquez sur **Edit schema** et assurez-vous que le schéma est récupéré du composant précédent. Sinon, cliquez sur **Sync Columns** pour récupérer le schéma du composant.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job docPays at 16:18 03/11/2009.
[statistics] connecting to socket on port 3916
[statistics] connected
AD|Andorra|[EUR]
AE|United Arab Emirates|[AED]
AF|Afghanistan|[AFN]
AG|Antigua and Barbuda|[XCD]
AI|Anguilla|[XCD]
AL|Albania|[ALL]
AM|Armenia|[AMD]
AN|Netherlands Antilles|[ANG]
AO|Angola|[AOA]
AR|Argentina|[ARS]
AS|American Samoa|[USD]
AT|Austria|[EUR]
AU|Australia|[AUD]
AW|Aruba|[AWG]
```

La liste des différents pays, ansi que leur code et devise est retournée dans la console de la vue **Run**.

tMDMOutput

Propriétés du tMDMOutput

Famille de composant	Talend MDM	
Fonction	Le composant **tMDMOutput** écrit des données maître dans un hub MDM (référentiel XML).	
Objectif	Ce composant écrit des données maître sur le serveur MDM.	
Basic settings	*Property Type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il devient automatiquement built-in. Cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	XML Field	Sélectionnez le nom de la colonne dans laquelle vous souhaitez écrire les données XML.
	URL	Saisissez l'URL d'accès au serveur MDM.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur du serveur MDM. Cet utilisateur doit avoir le bon rôle dans MDM, c'est-à-dire qu'il puisse se connecter via un Job ou tout autre appel de service Web. Pour plus d'informations, consultez votre guide **Talend Master Data Management Administrator Guide** (en anglais).
	Version	Saisissez le nom de la version des données maître à laquelle vous souhaitez vous connecter, en fonction de vos droits utilisateur. Laissez ce champ vide pour afficher la version par défaut des données maître.

	Data Model	Saisissez le nom du modèle de données en comparaison duquel les données à écrire seront validées.
	Data Container	Saisissez le nom du conteneur de données dans lequel vous souhaitez écrire les données maître.
	Return Keys	**Columns corresponding to IDs in order** : dans l'ordre séquentiel, définissez les colonnes de sortie qui stockeront les valeurs des clés retournées (clés primaires) des éléments à créer.
	Is Update	Cochez cette case pour mettre à jour les champs modifiés. Si vous ne cochez pas cette case, la totalité des champs sera remplacée par les champs modifiés.
	With Report	Cochez cette case pour ajouter les actions effectuées à un rapport de modifications. **Source Name** : Saisissez, entre guillemets, le nom de l'application à partir de laquelle les modifications sont apportées. **Enable verification by "before saving" transformer** : cochez cette case pour vérifier l'enregistrement qui vient d'être ajouté avant qu'il ne soit enregistré.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Rejects**.
Advanced settings	*Extended Output*	Cochez cette case pour commiter les données maître par lot. **Rows to commit** : saisissez le nombre de lignes par lot.
	Configure Xml Tree	Ouvrez l'interface d'aide à la création de la structure XML des données maître à écrire.
	Group by	Sélectionnez la colonne à utiliser pour regrouper les données maître.
	Create empty element if needed	Cette case est cochée par défaut. Si le contenu de la colonne **Related Column** de l'interface permettant de créer la structure XML est nul, ou si aucune colonne n'est associée au noeud XML, cette option créera une balise ouvrante et une balise fermante aux endroits prévus.
	Advanced separator (for number)	Cochez cette case pour modifier les séparateurs utilisés par défaut dans les nombres : - **Thousands separator** : saisissez entre guillemets le séparateur à utiliser pour les milliers. - **Decimal separator** : saisissez entre guillemets le séparateur à utiliser pour les décimales.
	Generation mode	Sélectionnez le mode de génération que vous souhaitez utiliser en fonction de votre mémoire disponible : - **Fast but memory-consuming (Dom4J)**, le plus rapide mais consommant de la mémoire. - **Slow with no memory consumed**, le plus lent ne consommant pas de mémoire.

	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Utilisez ce composant pour écrire un fichier et séparer les champs à l'aide d'un séparateur spécifque.	

Scénario : Ecrire des données maître dans un hub MDM

Ce scénario décrit un Job à deux composants qui génère un enregistrement de données, transforme ces données en XML et les charge dans l'entité métier définie sur le serveur MDM .

Dans cet exemple, l'objectif est de charger une nouvelle agence dans l'entité métier *Agency*. Cette nouvelle agence doit comprendre un identifiant, un nom et une ville.

- A partir de la **Palette**, glissez les composants **tFixedFlowInput** de la famille **Misc**, et **tMDMOutput** de la famille **Talend MDM** dans l'espace de modélisation.
- Reliez-les via un lien de type **Row Main**.
- Double-cliquez sur le composant **tFixedFlowInput** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés.

- Dans la liste **Schema**, sélectionnez **Built-In** et cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour décrire la structure des données maître que vous souhaitez écrire dans le serveur MDM.

- Ajoutez trois colonnes de type **String** en cliquant sur le bouton **[+]**. Nommez ces colonnes *Id*, *Name* et *City*.
- Cliquez sur **OK** pour valider vos modifications et passer à l'étape suivante.
- Dans le champ **Number of rows**, saisissez le nombre de lignes que vous souhaitez générer.
- Dans la zone **Mode**, sélectionnez le mode **Use Single Table** pour ne générer qu'une seule table.
- Dans le tableau **Values**, saisissez entre guillemets, dans chaque champ **Value**, la valeur correspondant à chaque colonne du schéma.
- Dans l'espace de modélisation, double-cliquez sur le composant **tMDMOutput** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés.

- Dans la liste **Property Type**, sélectionnez **Built-In** et renseignez les champs manuellement. Si vous avez stocké les informations de connexion au MDM dans les métadonnées du Repository, sélectionnez **Repository** dans la liste et les champs suivants seront renseignés automatiquement.
- Dans la liste **Schema**, sélectionnez **Built-In** et cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour décrire la structure des données maître que vous souhaitez charger sur le serveur MDM.

Le composant **tMDMOutput** génère un document XML, l'écrit dans un champ de sortie et l'envoie au serveur MDM, afin que le schéma de sortie ait toujours une colonne *xml* en lecture seule.

- Cliquez sur **OK** pour passer à l'étape suivante.
  La liste **XML Field** dans l'onglet **Basic settings** est automatiquement renseignée avec la colonne *xml* de sortie que vous venez de créer.

- Dans le champ **URL**, saisissez entre guillemets l'URL de connexion au serveur MDM.
- Dans les champs **Username** et **Password**, saisissez l'identifiant et le mot de passe de connexion au serveur MDM.
- Dans le champ **Version**, saisissez entre guillemets le nom de la version des données maître à laquelle vous souhaitez accéder. Laissez le champ vide pour accéder à la version par défaut.
- Dans le champ **Data Model**, saisissez entre guillemets le nom du modèle de données par rapport auquel vous souhaitez valider les données maître que vous souhaitez écrire.
- Dans le champ **Data Container**, saisissez entre guillemets le nom du conteneur de données dans lequel les données maître doivent être écrites.
- Dans la vue **Component**, cliquez sur l'onglet **Advanced settings** pour paramétrer les paramètres avancées du composant.

- Cochez la case **Extended Output** si vous souhaitez commiter les données maître par lot. Vous pouvez spécifier le nombre de lignes par lot dans le champ **Rows to commit**.
- Cliquez sur le bouton **[...]** du champ **Configure Xml Tree** pour ouvrir l'éditeur du composant.

- Dans la zone **Link target** à droite, cliquez dans le champ **Xml Tree** et remplacez **rootTag** par le nom de l'entité métier dans laquelle vous souhaitez insérer l'enregistrement de données, *Agency* dans cet exemple.
- Dans la zone **Linker source**, sélectionnez les trois colonnes de votre schéma et glissez les sur le noeud **Agency**.

La boîte de dialogue **[Selection]** s'ouvre.

- Sélectionnez **Create as sub-element of target node** pour que vos trois colonnes correspondent à trois sous-éléments du noeud **Agency** en XML. Cliquez sur **OK**.

- Dans la zone **Link target** cliquez-droit sur l'élément que vous souhaitez définir comme élément sur lequel effectuer une boucle, puis sélectionnez **Set as Loop Element** dans le menu contextuel. Dans cet exemple, **City** est l'élément d'itération.

- Cliquez sur **OK** pour valider vos changements et fermer l'éditeur.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

Le nouvel enregistrement de données est inséré dans l'entité métier *Agency*, dans le conteneur de données *DStar*, sur le serveur MDM. Cet enregistrement de données contient, comme défini dans le schéma, l'ID de l'agence, son nom et sa ville. La clé primaire de l'enregistrement inséré est également générée et retrouvée.

tMDMReceive

Propriétés du tMDMReceive

Famille de composant	Talend MDM	
Fonction	Le composant **tMDMReceive** reçoit un enregistrement MDM provenant de triggers MDM ou de processus MDM.	
Objectif	Ce composant décode un paramètre de contexte comprenant des données MDM XML et les transforme en schéma plat.	
Basic settings	*Property Type*	Peut être Built-in ou Repository.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Notez que si vous modifiez le schéma, il passera automatiquement en mode **Built-In**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	XML Record	Saisissez le paramètre de contexte vous permettant de récupérer les dernières modifications apportées au serveur MDM. Pour plus d'informations à propos de la création et de l'utilisation des paramètres de contexte, consultez *Définir les variables à partir de la vue Contexts* du **Guide Utilisateur** **Talend Open Studio**.
	XPath Prefix	Si nécessaire, sélectionnez dans la liste l'expression XPath de boucle, qui est la concaténation du préfixe et du chemin de la boucle. `/item` : sélectionnez ce préfixe XPath lorsque le composant reçoit l'enregistrement d'un processus, car les processus encapsulent l'enregistrement dans un élément "item" uniquement. /exchange/item : sélectionnez ce préfixe XPath lorsque le composant reçoit l'enregistrement d'un déclencheur, car les déclencheurs encapsulent l'enregistrement dans un élément 'item" dans un élément "exchange".
	Loop XPath query	Noeud de l'arborescence XML sur lequel est basé la boucle.

	Mapping	**Column** : reflète le schéma tel qu'il est défini dans l'éditeur Edit Schema. **XPath query** : Saisissez les champs à extraire de la structure XML d'entrée. **Get Nodes** : Cochez cette case pour ajouter les balises XML entournant les données à récupérer.
	Limit	Nombre maximum de lignes à traiter. Si Limit = 0, aucune ligne n'est lue ni traitée.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreur, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant est un composant de sortie. Il nécessite un flux de sortie.	

Scénario associé

Aucun scénario n'est disponible pour ce composant.

tMDMRouteRecord

Propriétés du tMDMRouteRecord

Famille de composant	Talend MDM	
Fonction	Le composant **tMDMRouteRecord** soumet la clé primaire d'un enregistrement stocké dans votre hub MDM (Repository) à l'**Event Manager**, afin que ce gestionnaire déclenche le ou les processus attendu(s) selon certaines conditions que vous devez définir dans les pages des processus et des déclencheurs. Pour plus d'informations concernant l'**Event Manager** et les processus MDM, consultez le Guide **Talend Master Data Management Administrator Guide** (en anglais).	
Objectif	Ce composant permet à l'**Event Manager** d'identifier les modifications apportées à vos données, pour que les actions corrélées puissent être déclenchées.	
Basic settings	*URL*	Saisissez l'URL d'accès au serveur MDM.
	Username et *Password*	Informations d'authentification de l'utilisateur du serveur MDM.
	Version	Saisissez le nom de la version des données maître à laquelle vous souhaitez vous connecter, en fonction de vos droits utilisateur. Laissez ce champ vide pour afficher la version par défaut des données maître.
	Entity	Saisissez le nom de l'entité contenant les données que vous souhaitez effacer.
	Data Container	Saisissez le nom du conteneur de données contenant les données que vous souhaitez effacer.
	Entity Name	Saisissez le nom de l'entité métier contenant l'enregistrement que vous souhaitez faire lire à l'**Event Manager**.
	IDS	Spécifiez la (les) clé(s) primaire(s) de(s) l'enregistrement(s) que vous souhaitez faire lire à l'**Event Manager**.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Global Variables		**Number of Lines** : indique le nombre de lignes traitées. Disponible en tant que variable **After**. Retourne un entier. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job, page 146* du **Guide Utilisateur Talend Open Studio**.

Connections		Liens de sortie (d'un composant à un autre) : **Row** : Iterate. **Trigger** : Run If, OnComponentOk, OnComponentError, OnSubjobOk, OnSubjobError. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate. **Trigger** : Run If, OnComponentOk, OnComponentError, OnSubjobOk, OnSubjobError. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur Talend Open Studio**.
Utilisation	Ce composant est un composant de début. Il nécessite un flux de sortie.	

Scénario : Router un enregistrement vers l'Event Manager

Dans ce scénario, le composant **tMDMRouteRecord** est utilisé pour soumettre à l'**Event Manager** la clé primaire d'un enregistrement notifiant une mise à jour, afin que l'**Event Manager** déclenche un processus informant l'utilisateur de la mise à jour effectuée.

Talend MDM *est sensible à la casse, respectez donc les majuscules et les minuscules lorsque vous réalisez ce scénario.*

Prérequis du scénario

Les prérequis pour la réalisation de ce scénario sont les suivants :

- Un conteneur de données contenant différents enregistrements utilisant un modèle spécifique. Dans ce scénario, le conteneur de données est *Product*, et un enregistrement est inclus dans le conteneur, en suivant le modèle nommé *Product* :

Cette capture d'écran présente l'un des enregistrements produit stockés avec tous les attributs que vous pouvez voir.

Pour plus d'informations concernant la création d'un conteneur de données ou d'un modèle de données, consultez votre **Guide Talend Master Data Management Administrator Guide** (en anglais).

Pour plus d'informations concernant la création d'un enregistrement et l'accès à ses attributs, consultez le **Guide Talend MDM Web User Interface User Guide** (en anglais).

- Un Job, utilisé pour informer l'utilisateur de la mise à jour, déjà déployé sur votre serveur MDM. Dans ce scénario, le Job est appelé *message*, il n'utilise que le composant **tMsgBox**.
- Double-cliquez sur le composant afin d'afficher et configurer ses **Basic settings** :

- Dans le champ **Title**, saisissez "*Talend MDM*".
- Dans le champ **Message**, saisissez le message qui apparaîtra : *"A record is updated"*.

Pour plus d'informations concernant le composant **tMsgBox**, consultez la section *tMsgBox, page 1202*.

Pour plus d'informations sur comment déployer un Job sur le serveur MDM, consultez le **Guide Talend Master Data Management Administrator Guide** (en anglais).

Router un enregistrement afin de déclencher le processus correspondant

Ce scénario vous explique comment déclencher un processus à l'aide d'un composant **tMDMRouteRecord**.

- Connectez-vous à l'interface Web **Talend MDM** et cliquez sur **Browse Records**. Pour plus d'informations concernant la connexion à l'interface Web **Talend MDM** et l'ouverture de la vue **Browse Records**, consultez le **Guide Talend MDM Web User Interface User Guide** (en anglais).
- Dans le coin supérieur droit de la page Web, cliquez sur le bouton afin d'afficher le panneau **Actions**.
- Dans le panneau **Actions** de droite, sélectionnez le conteneur de données requis, ainsi que le modèle de données contenant le fichier à mettre à jour. Dans ce scénario, le conteneur de données et le modèle de données sont tous les deux appelés *Product*.
- Cliquez sur **Save** afin de sauvegarder le conteneur et le modèle de données sélectionnés.
- Dans la vue **Browse Records**, sélectionnez l'entité souhaitée. Dans cet exemple, sélectionnez *Product*.

- Cliquez sur **Search** pour ouvrir la liste des enregistrements dans la partie inférieure de la page Web.

- Double-cliquez sur l'un des enregistrements produit afin d'afficher ses attributs dans une nouvelle vue dédiée à ce produit. Par exemple, ouvrez le produit *Talend Mug* ayant l'Id unique *231035938*.

- Dans cette vue, modifiez la valeur de l'un des attributs. Par exemple, mettez à jour ce produit. Rendez-le disponible en cochant la case **Availability**.
- Cliquez sur **Save** afin de valider la mise à jour.
- Ouvrez votre Studio **Talend MDM** et accédez au hub MDM (vue **MDM Server**). Pour plus d'informations concernant le lancement du Studio **Talend MDM** et sa connexion au hub MDM, consultez le **Guide Talend Master Data Management Administrator Guide** (en anglais).

- Sous le noeud **Job Repository** de la vue **MDM Server**, cliquez-droit sur le Job *message*.
- Dans le menu contextuel, sélectionnez **Generate Talend Job Caller Process**. Le processus utilisé pour appeler ce Job est généré et s'affiche dans le répertoire **Event Management** > **Process**.

- Sous le noeud **Event Management**, cliquez-droit sur **Trigger**.
- Dans le menu contextuel, sélectionnez **New**.
- Dans l'assistant **New Trigger**, nommez le déclencheur *TriggerMessage*.

- Cliquez sur **OK** afin d'ouvrir la vue du nouveau déclencheur dans l'espace de travail de votre studio.
- Dans la vue du déclencheur, configurez le déclencheur afin qu'il lance le processus appelant le Job *message* une fois la mise à jour effectuée.

- Dans le champ **Description**, saisissez *Trigger that calls the Talend Job: message_0.1.war*.
- Dans le champ **Entity**, sélectionnez l'entité métier sur laquelle vous souhaitez déclencher le processus, ou saisissez son nom. Dans cet exemple, son nom est *Update*.
- Dans le champ **Service JNDI Name**, sélectionnez **callprocess** dans la liste déroulante.
- Dans le champ **Service Parameters**, complétez la définition des paramètres en saisissant la valeur : *CallJob_message_0.1.war*. Cette valeur est le nom du processus à appeler, que vous pouvez trouver dans le dossier **Event Management** > **Process** dans la vue **MDM server**.
- Dans la zone **Trigger xPath Expressions**, cliquez sur le bouton sous le tableau, afin d'ajouter une ligne XPath.
- Dans la ligne nouvellement ajoutée, cliquez sur le bouton **[...]** pour ouvrir une boîte de dialogue dans laquelle vous pouvez sélectionner l'entité ou l'élément sur laquelle/lequel vous souhaitez définir des conditions. Dans cet exemple, son nom est *Update/OperationType*.

- Dans la colonne **Value**, saisissez une valeur pour cette ligne. Dans cet exemple, la valeur est *UPDATE*.
- Dans la colonne **Condition Id**, saisissez un identifiant unique pour la condition que vous souhaitez définir, par exemple *C1*.
- Dans la zone **Conditions**, saisissez la requête que vous souhaitez exécuter sur les données en utilisant l'ID de la condition ID *C1* défini plus tôt.
- Appuyez sur **Ctrl+S** pour sauvegarder le déclencheur.
- Dans la vue **MDM server**, double-cliquez sur **Data container** > **system** > **UpdateReport** pour ouvrir la vue **Data Container Browser UpdateReport**. Le rapport **UpdateReport** contient les traces complètes de toutes les actions de création, de mise à jour ou de suppression (create, update ou delete) effectuées sur les données maître.

- A côté du champ **Entity** de cette vue, cliquez sur le bouton pour rechercher tous les enregistrements concernant les actions dans le rapport **UpdateReport**. Notez que l'entité **Update** ne signifie pas nécessairement que l'action enregistrée est une mise à jour, c'est simplement le nom de l'entité définie par le modèle de données du rapport **UpdateReport** pouvant contenir différentes actions comme *create*, *delete*, et *update*.
- Le dernier enregistrement correspond aux modifications apportées à l'enregistrement produit au début du scénario. La clé primaire de cet enregistrement est *genericUI.1283244014172*. Cet enregistrement sera routé vers l'**Event trigger**.
- Dans la barre de menu du studio, cliquez sur **Window** > **Perspective** > **Design workspace** afin de créer le Job routant l'enregistrement.
- Dans la perspective **Design workspace**, créez un Job et nommez-le *RouteRecord*.
- Pour ce faire, cliquez-droit sur **Job Designs**, dans la vue **Repository**. Dans le menu contextuel, sélectionnez **Create Job**.
- Un assistant s'ouvre. Dans le champ **Name**, saisissez *RouteRecord*, puis cliquez sur **Finish**.
- Déposez le composant **tMDMRouteRecord** de la **Palette** dans l'espace de modélisation graphique.
- Double-cliquez sur ce composant afin d'ouvrir sa vue **Component**.

- Dans le champ **URL**, saisissez l'adresse de votre serveur MDM. *http://localhost:8080/talend/TalendPort, dans cet exemple*.
- Dans les champs **Username** et **Password**, saisissez respectivement votre identifiant et votre mot de passe de connexion.
- Dans le champ **Data Container** saisissez le nom du conteneur de données contenant l'enregistrement que vous souhaitez router vers l'**Event Manager**. Dans cet exemple, son nom est *UpdateReport*.
- Dans le champ **Entity Name**, saisissez le nom de l'entité à laquelle appartient l'enregistrement que vous souhaitez router. Dans cet exemple, son nom est *Update*.
- Dans la zone **IDS**, cliquez sur le bouton **[+]** sous le tableau, afin d'ajouter une ligne.
- Dans cette nouvelle ligne, renseignez la clé primaire de l'enregistrement devant être routé vers l'**Event manager**, c'est-à-dire *genericUI.1283244014172*, comme vous l'avez vu dans le rapport **Data Container Browser UpdateReport**.
- Appuyez sur **F6** pour exécuter le Job. L'**Event manager** appelle le processus devant exécuter le Job *message* et générer la boîte de dialogue informant l'utilisateur de la mise à jour effectuée.

Le composant **tMDMRouteRecord** est utilisé pour soumettre à l'**Event Manager** la clé primaire d'un enregistrement notifiant une mise à jour. L'**Event Manager** analyse le fichier et

vérifie que les conditions définies dans la vue de configuration du déclencheur *Trigger's message* sont remplies, puis appelle le processus lançant le Job *Message*. Ce Job informe l'utilisateur de la mise à jour effectuée en faisant apparaître une fenêtre pop-up.

tMDMSP

Propriétés du tMDMSP

Famille de composant	Talend MDM	
Fonction	Le composant **tMDMSP** appelle une procédure stockée du hub MDM.	
Objectif	**tMDMSP** permet de centraliser des requêtes multiples ou complexes dans une base de données et de les appeler plus facilement.	
Basic settings	*Schema* et *Edit Schema*	Dans une procédure stockée, le schéma est un paramètre d'entrée. Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement.Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	URL	Saisissez l'URL du serveur MDM.
	Username et *Password*	Informations d'authentification sur l'utilisateur de base de données.
	Version	Saisissez le nom de la version des données maître à laquelle vous souhaitez vous connecter, en fonction de vos droits utilisateur. Laissez ce champ vide pour afficher la version par défaut des données maître.
	Data Container	Saisissez le nom du conteneur de données contenant les données que vous souhaitez effacer.
	SP Name	Saisissez le nom exact de la Procédure Stockée.
	Parameters (in order)	Cliquez sur le bouton **[+]** et sélectionnez les colonnes d'entrée (**Input Columns**) requises pour les procédures. Le schéma SP peut contenir plus de colonnes qu'il n'y a de paramètres utilisés dans la procédure.

Connections		Liens de sortie (d'un composant à l'autre) : **Row** : Main **Trigger** : Run if, OnComponentOk, OnComponentError, OnSubjobOk, OnSubjobError. Liens d'entrée (d'un composant à l'autre) : **Row** : Main, Iterate; Trigger : Run if, OnComponentOk, OnComponentError, OnSubjobOk, OnSubjobError Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** **Talend Open Studio**.
Utilisation	Ce composant est un composant intermédiaire. Il peut être utilisé comme composant de début mais aucun paramètre d'entrée n'est requis pour la procédure à appeler.	
Limitation	N/A	

Scénario : Exécuter une procédure stockée dans le hub MDM

Le but du Job suivant est de calculer le prix total de chaque produit enregistré dans votre interface Web MDM.

Ce Job va générer des paramètres pour l'exécution d'une procédure stockée dans le hub MDM, puis extraire les données souhaitées du résultat retourné au format XML et présenter les données extraites dans le studio.

Les produits dont le prix est à traiter sont listés dans votre interface Web MDM.

La procédure stockée devant être exécutée peut être trouvée dans le nœud **Stored Procedure** de l'arborescence **MDM Server** et apparaît comme suit :

Pour plus d'informations concernant la procédure stockée dans le serveur MDM, consultez le Guide **Talend Master Data Management Administrator Guide** (en anglais).

Pour créer ce Job, procédez comme suit :

- Glissez-déposez les composants suivants utilisés dans cet exemple : **tFixedFlowInput**, **tMDMSP**, **tExtractXMLField**, **tLogRow**.
- Connectez les composants à l'aide de liens **Row Main**.
- Le composant **tFixedFlowInput** sert à générer la gamme de prix vous intéressant pour ce calcul. Dans cet exemple, définissez 10 comme minimum et 17 comme maximum afin de prendre en compte tous les produits.
- Double-cliquez sur le composant tFixedFlowInput afin d'ouvrir sa vue **Component**.
- Dans la vue **Component**, cliquez sur le bouton **[...]** à côté du champ **Edit schema** afin d'ouvrir l'éditeur de schéma de ce composant.
- Dans l'éditeur de schéma, ajoutez les deux paramètres *min* et *max* utilisés pour définir la gamme de prix.

- Cliquez sur **OK** afin de valider cette modification.
- Dans la table **Values** dans la zone **Mode** de la vue **Component**, les deux paramètres *min* et *max* que vous avez définis dans l'éditeur de schéma du composant s'affichent.
- Dans la colonne **Value** de la table **Values**, saisissez le nombre 10 pour le paramètre *min* et 17 pour le paramètre *max*.

- Double-cliquez sur le composant **tMDMSP** afin d'ouvrir sa vue **Component**.

- Dans le champ **URL** de la vue **Component**, saisissez l'adresse du serveur MDM, dans cet exemple : *http://localhost:8080/talend/TalendPort*.
- Dans les champs **Username** et **Password**, saisissez les données d'authentification, dans cet exemple : *admin* et *talend* respectivement.
- Dans les champs **Data Container** et **Procedure Name**, saisissez les noms exacts du conteneur de données *Product* et de la procédure stockée *PriceAddition*.
- Sous la table **Parameters (in order)**, cliquez deux fois sur le bouton **[+]** afin d'ajouter deux lignes à cette table.
- Dans la table **Parameters (in order)**, cliquez sur chaque ligne ajoutée et, dans la liste déroulante, sélectionnez le paramètre *min* pour l'une et le paramètre *max* pour l'autre.
- Double-cliquez sur le composant **tExtractXMLField** afin d'ouvrir sa vue **Component**.

- Dans la vue **Component**, cliquez sur le bouton **[...]** situé à côté du champ **Edit schema** afin d'ouvrir l'éditeur de schéma du composant.

- Dans l'éditeur de schéma, ajoutez deux colonnes afin de définir la structure des données sortantes. Ces deux colonnes sont *name* et *sum*. Elles représentent respectivement le nom et le prix total de chaque type de produit enregistré dans l'interface Web.

- Cliquez sur **OK** afin de valider la configuration. Les deux colonnes s'affichent dans la table **Mapping** de la vue **Component**.
- Dans le champ **Loop XPath Query**, saisissez le nœud de l'arborescence XML sur lequel effectuer une boucle. Dans l'exemple, le nœud est */result* comme vous pouvez lire dans le code de la procédure : `return <result><Name>{$d}</Name><Sum>{sum($product/Price)}</Sum></result>`.
- Dans le champ **XPath Query** de la table **Mapping**, saisissez le nom exact du nœud sur lequel effectuer la boucle. Pour la colonne *name*, saisissez "*/result/Name*", pour mapper les noms du produit. Pour la colonne *sum*, saisissez "*/result/Sum*", pour extraire les prix totaux.
- Double-cliquez sur le composant **tLogRow** afin d'ouvrir sa vue **Component**.

- Synchronisez le schéma avec celui du composant précédant (en cliquant sur le bouton **Sync columns**).

- Dans la zone **Mode**, sélectionnez l'option **Table (print values in cells of a table)** pour une meilleure lecture.
- Appuyez ensuite sur **F6** afin d'exécuter le Job.
- Les données sortantes s'affichent dans la console de la vue **Run**.

```
Starting job feature14026_tMDMSP at 17:29 25/08/2010.

[statistics] connecting to socket on port 3487
[statistics] connected
.-------------------------+-----------------.
|                tLogRow_1                  |
|=------------------------+-----------------=|
|name                     |sum              |
|=------------------------+-----------------=|
|Talend Golf Shirt        |16.99            |
|Talend Jr. Spaghetti Tank|16.99            |
|Talend Large Mug         |11.99            |
|Talend Mug               |94.92999999999999|
'-------------------------+-----------------'

[statistics] disconnected
Job feature14026_tMDMSP ended at 17:29 25/08/2010. [exit
code=0]
```

La sortie liste les quatre sortes de produit enregistrés dans l'interface Web MDM et le prix total pour chaque type de produit.

tMDMViewSearch

Propriétés du tMDMViewSearch

Famille de composant	Talend MDM	
Fonction	Le composant **tMDMViewSearch** sélectionne des enregistrements d'un hub MDM (Repository XML) en appliquant les filtres que vous avez créés dans un but précis. Les données récupérées ont une structure XML. Pour plus d'informations concernant la vue permettant de définir des critères de filtres, consultez le **Guide Talend Master Data Management Administrator Guide** (en anglais).	
Objectif	**tMDMViewSearch** permet de retrouver des enregistrements MDM d'un hub MDM.	
Basic settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement.Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Jobs. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	XML Field	Sélectionnez le nom de la colonne dans laquelle vous souhaitez écrire les données XML.
	URL	Saisissez l'URL du serveur MDM.
	Username et *Password*	Informations d'authentification sur l'utilisateur de base de données.
	Version	Saisissez le nom de la version des données maître à laquelle vous souhaitez vous connecter, en fonction de vos droits utilisateur. Laissez ce champ vide pour afficher la version par défaut des données maître.
	Data Container	Saisissez le nom du conteneur de données contenant les données que vous souhaitez effacer.
	View Name	Saisissez le nom de la vue dont les filtres seront appliqués pour traiter les enregistrements.

	Operations	Remplissez ce tableau afin de créer une clause WHERE. Les paramètres à définir sont : - **XPath** : définissez le chemin de l'expression afin de sélectionner le noeud XML sur lequel appliquer les filtres. - **Functions** : sélectionnez un opérateur dans la liste déroulante, comme **Contains**, **Starts with**, **Equals**, etc. - **Value** : saisissez la valeur que vous souhaitez récupérer. - **Predicate** : sélectionnez le prédicat permettant de combiner les conditions de filtres de différentes manières. Le prédicat peut être **none**, **or**, **and**, **exactly**, etc. *Les paramètres sont sensibles à la casse.*
	Order (One Row)	Remplissez ce tableau afin de choisir l'ordre de présentation des enregistrements récupérés. Les paramètres à définir sont : - **XPath** : définissez le chemin de l'expression afin de sélectionner le noeud XML sur lequel appliquer le tri. - **Order** : sélectionnez l'ordre de présentation, soit **asc** (ascendant) ou **desc** (descendant). *Les paramètres sont sensibles à la casse.* *Pour le moment, seule la première ligne créée dans le tableau Order est valide.*
	Spell Threshold	Configuez cette option sur -1 afin de désactiver le seuil. Ce seuil est utilisé pour décider du niveau de tolérance des erreurs orthographiques.
	Skip Rows	Renseignez le nombre de lignes à ignorer afin de spécifier à partir de quelle ligne doit commencer le processus. Par exemple, si vous saisissez *8*, le processus commencera à la neuvième ligne.
	Max Rows	Saisissez le nombre maximal de lignes à traiter. Si le champ **Limit** est paramétré sur *0*, aucune ligne ne sera lue ou traitée. Par défaut, la valeur de ce champ est *-1*, ce qui signifie qu'aucune limite n'est définie.
Utilisation	Utilisez ce composant pour récupérer des enregistrements spécifiques.	
Global Variables		**Number of Lines** : Indique le nombre de lignes traitées. Disponible en tant que variable **After**. Retourne un entier. Pour plus d'informations concernant les variables, consultez *Utiliser les variables dans un Job* dans le **Guide Utilisateur Talend Open Studio**.

Connections		Liens de sortie (d'un composant à un autre) : **Row** : Iterate **Trigger** : Run if, OnComponentOk, OnComponentError, OnSubjobOk, OnSubjobError. Liens d'entrée (d'un composant à un autre) : **Row** : Iterate; **Trigger** : Run if, OnComponentOk, OnComponentError, OnSubjobOk, OnSubjobError. Pour plus d'informations concernant les connexions, consultez *Types de connexions* dans le **Guide Utilisateur** Talend Open Studio.
Limitation	N/A	

Scénario : Récupérer des enregistrements d'un hub MDM hub via une vue existante

Ce scénario décrit un Job comprenant deux composants récupérant un enregistrement de données ayant une structure XML.

Dans cet exemple, vous allez sélectionner les informations concernant le T-shirt de l'entité *Product* via la vue *Browse_items_Product* créée dans Talend Open Studio. Chaque enregistrement de l'entité contient les détails définis comme critères de filtre : *Id*, *Name*, *Description* et *Price* (ID, Nom, Description et Prix).

- A partir de la **Palette**, déposez un composant **tMDMViewSearch** et un **tLogRow** dans l'espace de modélisation graphique.
- Reliez les composants à l'aide d'un lien **Row Main**.
- Double-cliquez sur le **tMDMViewSearch** afin d'ouvrir sa vue **Basic settings**, dans l'onglet **Component** et de définir les propriétés du composant.

- Dans la liste **Schema**, sélectionnez **Built-In** puis cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour ouvrir une boîte de dialogue dans laquelle vous pouvez définir la structure des données XML que vous souhaitez écrire.

- Cliquez sur le bouton **[+]** pour ajouter une colonne de type **String**. Nommez-la *Tshirt*.
- Cliquez sur **OK** pour valider la modification et passer à l'étape suivante.
- Dans le champ **XML Field**, sélectionnez *Tshirt* comme colonne dans laquelle vous souhaitez écrire les données récupérées.

- Dans le champ **URL**, saisissez l'adresse de votre serveur MDM et renseignez respectivement votre identifiant et votre mot de passe de connexion dans les champs **Username** et **Password**. Dans cet exemple, utilisez l'URL par défaut, puis saisissez *admin* comme identifiant et mot de passe.
- Dans le champ **Data Container**, saisissez le nom du conteneur de données : *Product*.
- Dans le champ **View Name**, saisissez le nom de la vue : *Browse_item_Product*.
- Sous le tableau **Operations**, cliquez sur le bouton **[+]** pour ajouter une ligne.
- Dans le tableau **Operations**, définissez la colonne **XPath** comme *Product/Name*, ce qui signifie que l'opération de filtre sera effectuée sur le noeud *Name*. Sélectionnez ensuite *Contains* dans la colonne **Function** puis saisissez *Tshirt* dans la colonne **Value**.
- Sous le tableau **Order (One Row)**, cliquez sur le bouton **[+]** afin d'ajouter une ligne.
- Dans le tableau **Order (One Row)**, défininissez la colonne **XPath** comme *Product/Id* et sélectionnez l'ordre **asc** dans la colonne **Order**.
- Dans l'espace de modélisation graphique, cliquez sur le **tLogRow** afin d'ouvrir sa vue **Basic settings** et définir ses propriétés.

- A côté du bouton **[...]** utilisé pour éditer le schéma, cliquez sur **Sync columns** pour retrouver le schéma du composant précédent.
- Appuyez sur **F6** afin d'exécuter le Job.

```
Starting job feature14031_tMDMViewSearch at 17:37 10/08/2010.

[statistics] connecting to socket on port 3920
[statistics] connected
<result>
    <Id>1</Id>
    <Name>Tshirt-V</Name>
    <Description>green</Description>
    <Price>16.00 </Price>
</result>
<result>
    <Id>2</Id>
    <Name>Tshirt-G</Name>
    <Description>Green</Description>
    <Price>17.00</Price>
</result>
[statistics] disconnected
Job feature14031_tMDMViewSearch ended at 17:37 10/08/2010.
[exit code=0]
```

Dans la console de la vue **Run**, vous pouvez voir les enregistrements *Tshirt* retournés, avec leur structure XML, triés en ordre ascendant.

tStewardshipTaskInput

Ce composant est disponible dans la **Palette** *de* ***Talend Open Studio*** *mais vous pourrez l'utiliser uniquement si vous avez souscrit à l'édition correspondante de* ***Talend Master Data Management****.*

Propriétés du tStewardshipTaskInput

Famille de composant	Talend MDM	
Fonction	Le composant **tStewardshipTaskInput** lit des ensembles de données maître stockés dans la base de données de la console de stewardship sous forme de tâches. Ce composant peut récupérer des tâches selon certains critères de recherche ou sans critère de recherche, à condition que le schéma de sortie soit le même pour toutes les tâches récupérées. Afin de mieux comprendre l'objectif de ce composant, consultez le **Guide Talend Data Stewardship Console User Guide** (en anglais).	
Objectif	Ce composant lit des données maître dans la console de stewardship et permet le traitement de ces données, c'est-à-dire d'utiliser tout composant **Talend** de sortie et d'écrire le données récupérées à partir de la base de données de stewardship dans l'application cible ou dans la base de données.	
Basic Settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Si vous modifiez le schéma, il devient automatiquement **Built-In**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Host	Adresse IP du serveur de la base de données de la console de stewardship.
	Port	Numéro du port d'écoute du serveur.
	Type	Si nécessaire, sélectionnez le type des tâches que vous souhaitez lire : **Resolution** : les tâches de résolution de données représentent les résultats des processus de mise en correspondance des données effectués sur les données de différentes sources. **Data** : les tâches d'intégrité de données sont les résultats des processus d'intégrité de données effectués sur les données. Pour plus d'informations concernant les types de tâches et la gestion des tâches, consultez le **Guide Talend Data Stewardship Console User Guide**(en anglais).

	Owner	Au besoin, saisissez le nom du propriétaire de la tâche. Cela filtrera les tâches à lire par nom de propriétaires.
	Tag	Si nécessaire, saisissez le nom de la catégorie du tag associé aux tâches que vous souhaitez lire. Pour plus d'informations, consultez le **Guide Talend Data Stewardship Console User Guide**(en anglais).
	Start Date/End Date	Au besoin, saisissez une plage de dates de création de tâches. Vous pourrez lire les tâches comprises das cet intervalle.
	Star ranking	Au besoin, sélectionnez le nombre d'étoiles, de 0 à 5, affectée(s) aux tâches, comme un classement numérique mettant en valeur l'importance des tâches. Ce classement permet de filtrer les tâches selon leur importance.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant..
Utilisation	Ce composant est un composant d'entrée. Il nécessite un flux de sortie.	

Scénario : Lire des données dans la base de données de la console de stewardship

Ce scénario décrit un Job a deux composants qui lit un ensemble de données à partir de la base de données de **Talend Data Stewardship Console**. Vous pouvez utiliser n'importe quel composant **Talend** de sortie et écrire les données récupérées de la base de données de stewardship dans l'application ou la base de données.

- Les données maître filtrées sont récupérées et affichées dans la console de log.

- Déposez les composants **tStewardshipTaskInput** et **tLogRow** de la **Palette** dans l'espace de modélisation graphique.
- Connectez les deux composants à l'aide d'un lien **Row Main**.
- Double-cliquez sur le **tStewardshipTaskInput** afin d'ouvrir sa vue **Basic settings** et définir ses propriétés.

- Dans la liste **Schema**, sélectionnez **Built-In** et cliquez sur le bouton **[...]** à côté du champ **Edit schema** afin d'ouvrir une boîte de dialogue.
  Vous pouvez définir la structure des données que vous souhaitez lire dans la base de données **Talend Data Stewardship Console**.

- Les données sont collectées des trois colonnes d'entrée définies : *Firstname*, *Lastname* et *DOB* (prénom, nom et date de naissance).
  Les données sont également collectées à partir de toutes les colonnes par défaut correspondant à la liste des tâches dans **Talend Data Stewardship Console**.
- Cliquez sur **OK** pour fermer la boîte de dialogue et passer à l'étape suivante.
- Dans le champ **Host**, saisissez entre guillemets l'adresse IP du serveur de la base de données de la console de stewardship.
- Dans le champ **Port**, saisissez entre guillemets le numéro du port d'écoute du serveur de la base de données.
- Dans les champs **Username** et **Password**, saisissez votre identifiant et votre mot de passe de connexion au serveur MDM.

- Dans la liste **Type**, sélectionnez le type de tâches que vous souhaitez récupérer de la console de stewardship : **Resolution** ou **Data**. Dans cet exemple, seules les tâches de résolution sont récupérées.
  Pour plus d'informations concernant les types de tâches, consultez le **Guide** **Talend Data Stewardship Console** **User Guide** (en anglais).
- Dans le champ **Owner**, saisissez entre guillemets le nom du propriétaire de la tâche, c'est-à-dire l'utilisateur auquel est assignée la tâche, *Administrator* dans cet exemple.

  Une tâche peut être assignée à un utilisateur spécifique soit à partir de la vue **Basic settings** du composant **tStewardshipTaskOutput**, soit directement à partir de la console de stewardship par un administrateur. Pour plus d'informations, consultez *tStewardshipTaskOutput, page 1428* et le **Guide** **Talend Data Stewardship Console** **User Guide** (en anglais).

- Dans le champ **Tag**, saisissez le nom de la catégorie du tag associé aux tâches que vous souhaitez lire. Ce champ n'est pas utilisé dans ce scénario.
  Pour plus d'informations, consultez le **Guide** **Talend Data Stewardship Console** **User Guide** (en anglais).
- Dans les champs **Start Date** et **End Date** fields, saisissez entre guillemets une plage de dates de création de tâches durant laquelle vous souhaitez lire les tâches. Ce champ n'est pas utilisé dans ce scénario.
- Dans le champ **Star ranking**, sélectionnez dans la liste le nombre d'étoiles, de 0 à 5, à assigner aux tâches dans la console de stewardship. Cela permet de filtrer les tâches à récupérer selon leur nombre d'étoiles.
  Dans cet exemple, sélectionnez *2* dans la liste. Toutes les tâches ayant deux étoiles ou plus seront récupérées.

  Si vous sélectionnez *All* dans la liste, vous récupérerez toutes les tâches, quel que soit leur nombre d'étoiles.

- Dans le champ **Limit**, saisissez un nombre afin de limiter les tâches à récupérer dans la base de données de stewardship.
- Dans l'espace de modélisation graphique, double-cliquez sur le composant **tLogRow** afin d'afficher sa vue **Basic settings** et définir ses propriétés.
- Cliquez sur **Edit Schema** pour ouvrir la boîte de dialogue du schéma et vérifier que le schéma a bien été reçu du composant précédent. Si ce n'est pas le cas, cliquez sur **Sync Columns**.
- Sauvegardez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job FetchTasks at 17:33 28/09/2010.

[statistics] connecting to socket on port 3359
[statistics] connected
.-------------------------------------------------------.
|                    #1. tLogRow_1                      |
+-----------------+-----------------------------------+
| key             | value                             |
+-----------------+-----------------------------------+
| Firstname       | null                              |
| Lastname        | null                              |
| DOB             | null                              |
| TASK_ID         | 8aaa94e42b57aefd012b5861e368006d  |
| TASK_TYPE       | Resolution                        |
| TASK_CREATEDBY  | administrator                     |
| TASK_CREATEDON  | 2010-09-28T14:47:11.000Z          |
| TASK_OWNER      | administrator                     |
| TASK_STATUS     | new                               |
| TASK_STARS      | 2                                 |
+-----------------+-----------------------------------+
.-------------------------------------------------------.
|                    #2. tLogRow_1                      |
+-----------------+-----------------------------------+
| key             | value                             |
+-----------------+-----------------------------------+
| Firstname       | Lily                              |
| Lastname        | Massy                             |
| DOB             | 1-10-1982                         |
| TASK_ID         | 8aaa94e42b57aefd012b58b1fe5500fa  |
| TASK_TYPE       | Resolution                        |
| TASK_CREATEDBY  | administrator                     |
| TASK_CREATEDON  | 2010-09-28T16:14:41.000Z          |
| TASK_OWNER      | administrator                     |
| TASK_STATUS     | resolved                          |
| TASK_STARS      | 4                                 |
+-----------------+-----------------------------------+
.-------------------------------------------------------.
|                    #3. tLogRow_1                      |
+-----------------+-----------------------------------+
| key             | value                             |
+-----------------+-----------------------------------+
| Firstname       | Martin                            |
| Lastname        | Richards                          |
| DOB             | 1-10-1972                         |
| TASK_ID         | 8aaa94e42b57aefd012b58b1feb3010a  |
| TASK_TYPE       | Resolution                        |
| TASK_CREATEDBY  | administrator                     |
| TASK_CREATEDON  | 2010-09-28T16:14:41.000Z          |
| TASK_OWNER      | administrator                     |
| TASK_STATUS     | resolved                          |
| TASK_STARS      | 3                                 |
+-----------------+-----------------------------------+
[statistics] disconnected
Job FetchTasks ended at 17:33 28/09/2010. [exit code=
```

Le composant **tStewardshipTaskInput** a récupéré les ensembles de données stockés dans la base de données de la console de stewardship, sous forme de tâches selon les critères de recherche que vous utilisez. Le schéma de sortie est le même pour toutes les tâches récupérées.

tStewardshipTaskOutput

Ce composant est disponible dans la **Palette** *de* ***Talend Open Studio*** *mais vous pourrez l'utiliser uniquement si vous avez souscrit à l'édition correspondante de* ***Talend Master Data Management****.*

Propriétés du tStewardshipTaskOutput

Famille de composant	Talend MDM	
Fonction	Le composant **tStewardshipTaskOutput** écrit des données, sous forme de tâches, dans la base de données de **Talend Data Stewardship Console** et permet ainsi de lister les tâches dans la console de stewardship des données. Un "steward" autorisé peut intervenir pour effectuer la mise en correspondance des données listées ou pour vérifier que les données sont cohérentes et complètes. Pour mieux comprendre le fonctionnement de ce composant, consultez le **Guide Talend Data Stewardship Console User Guide** (en anglais).	
Objectif	Ce composant crée des tâches dans la base de données **Talend Data Stewardship Console**.	
Basic Settings	*Schema* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le **Repository**. Cliquez sur **Edit Schema** pour modifier le schéma. Si vous modifiez le schéma, il devient automatiquement **Built-In**.
		Built-in : Le schéma sera créé et conservé pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide Utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé. Voir également : *Paramétrer un schéma du Repository* du **Guide Utilisateur** de **Talend Open Studio**.
	Host	Adresse IP du serveur de la base de données de la console de stewardship.
	Port	Numéro du port d'écoute du serveur de la base de données.
	Username et *Password*	Saisissez les informations d'authentification de l'utilisateur pour la base de données de la console de stewardship.
	Type	Si nécessaire, sélectionnez le type des tâches que vous souhaitez lire : **Resolution** : les tâches de résolution de données représentent les résultats des processus de mise en correspondance des données effectués sur les données de différentes sources. **Data** : les tâches d'intégrité de données sont les résultats des processus d'intégrité de données effectués sur les données. Pour plus d'informations concernant les types de tâches et la gestion des tâches, consultez le **Guide Talend Data Stewardship Console User Guide** (en anglais).

	Created by	Saisissez le nom du créateur de la tâche. Les créateurs de tâches correspondent aux utilisateurs de l'interface Web de **Talend MDM**. Pour plus d'informations, consultez le **Guide Talend MDM Web User Interface User Guide** (en anglais).
	Owner	Saisissez le nom du propriétaire de la tâche. Les propriétaires de tâches correspondent aux utilisateurs de l'interface Web de **Talend MDM**. Pour plus d'informations, consultez le **Guide Talend MDM Web User Interface User Guide** (en anglais).
	Star	Saisissez le chiffre, entre 0 et 5, que vous souhaitez assigner aux tâches comme classement numérique, sous forme d'étoiles, mettant en valeur l'importance des tâches.
	Tag	Saisissez le nom de la catégorie du tag que vous souhaitez associer aux tâches que vous voulez écrire. *Les catégories de tag doivent avoir été créées précédemment dans la console de stewardship. Pour plus d' informations concernant la création de catégories de tag, consultez le* **Guide** ***Talend Data Stewardship Console*** **User Guide** *(en anglais).*
Tâches de résolution uniquement	*Looping column*	Sélectionnez une colonne dans le schéma d'entrée sur laquelle effectuer la boucle. Lorsque la valeur de la colonne sur laquelle est effectuée la boucle change, le composant ferme l'élément précédent (tâche) et en ouvre un nouveau (nouvelle tâche). La colonne sur laquelle est effectuée la boucle est généralement l'ID du groupe généré par le composant **tMatchGroup**.
	Source/Target selector	Sélectionnez une colonne dans le schéma d'entrée qui décidera si les enregistrements de tâches définis selon la colonne de boucle seront des enregistrements source ou cible.
	Source	Sélectionnez une colonne dans le schéma d'entrée.
Tâches de résolution uniquement	*Score*	Sélectionnez la colonne de score de correspondance dans le schéma d'entrée.
Tâches de résolution uniquement	*Weights*	Sélectionnez la colonne définissant la distance de correspondance pour chaque colonne du schéma d'entrée.

	Extra info	Si nécessaire, utilisez le bouton **[+]** pour ajouter une ou plusieurs ligne(s) pour toute information supplémentaire, que vous souhaitez ajouter à l'un des enregistrements source. Dans la colonne **Title**, saisissez les informations clés. Dans la colonne **Message**, saisissez les informations que vous souhaitez ajouter. Dans la colonne Column, cliquez dans la ligne ajoutée et sélectionnez la colonne source à laquelle vous souhaitez ajouter l'information supplémentaire. Le "steward" peut voir les informations ajoutées à chaque fois qu'il place sa souris sur la colonne d'enregistrement dans **Talend Data Stewardship Console**. Cette information lui permettra de prendre des décisions plus informées lors de la résolution de la tâche.
	Record column	Utilisez le bouton **[+]** afin d'ajouter autant de lignes que nécessaire, puis cliquez dans chacune des lignes et sélectionnez les colonnes du schéma d'entrée qui formeront l'enregistrement cible.
	Max tasks per commit	Saisissez le nombre maximal de tâches par commit.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant..
Utilisation	Ce composant écrit des enregistrements des données contenus dans des tâches. Ce composant nécessite un flux d'entrée.	

Scénario : Ecrire des enregistrements de données dans la base de données de la console de stewardship

Ce scénario décrit un Job à cinq composants générant des enregistrements de données sous forme de tâches et chargeant celles-ci dans la base de données de la console de stewardship.

Ces tâches nécessitent l'intervention d'un "steward" ayant l'autorisation de fusionner, de comparer, et de collecter les enregistrements de données contenus dans ces tâches. Pour plus d'informations, consultez le **Guide Utilisateur** **Talend Data Stewardship Console** (en anglais).

Dans ce scénario :

- Un composant **tFixedFlowInput** génère un flux de données d'entrée possédant cinq colonnes : *Source*, *Firstname*, *Lastname*, *DOB* (date of birth), et *PostalCode* (Source, prénom, nom, date de naissance et code postal). Ces données contiennent ddes problèmes telles que des doublons, des prénoms ou des noms mal écrits ou écrits différemment, des informations différentes pour le même client, etc.
- Un composant de qualité de données, le **tMatchGroup**, effectue des opérations de mise en correspondance des données de différentes sources définies dans la colonne d'entrée *Source*. Ce composant regroupe les colonnes de sortie par une valeur de bloc, afin d'optimiser l'opération de mise en correspondance et de comparer uniquement les enregistrements ayant la même valeur de bloc, la colonne *Source* dans ce scénario.
- Un composant **tMap** filtre le flux d'entrée et sépare les enregistrements de données uniques et les enregistrements de données ayant une distance de correspondance.

- Les enregistrements de données uniques sont affichés dans la console de la vue **Run**, via le composant **tLogRow**. Tous les autres enregistrements de données, ayant une distance de correspondance sont envoyés dans la base de données de **Talend Data Stewardship Console** via le composant **tStewardshipTaskOutput** et affichés dans la console de stewardship. Un “steward” autorisé peut intervenir pour fusionner les enregistrements de données ayant des distances de correspondance.

- Déposez les composants suivants de la **Palette** dans l’espace de modélisation graphique : **tFixedFlowInput**, **tMatchGroup**, **tMap**, **tStewardshipTaskOutput** et **tLogRow**.
- Reliez les trois premiers composants à l’aide de liens **Main**.
- Double-cliquez sur le **tFixedFlowInput** afin d’afficher sa vue **Basic settings** et définir ses prorpiétés.
  Le composant **tFixedFlowInput** génère un flux de données d’entrée comprenant cinq colonnes : *Source*, *Firstname*, *Lastname*, *DOB* (date of birth), et *PostalCode* (Source, prénom, nom, date de naissance et code postal). Ces données contiennent des problèmes comme des doublons, des prénoms ou noms mal écrits ou écrits différemment, des informations différentes pour le même client, etc.

- Double-cliquez sur le composant **tMatchGroup** afin d'afficher sa vue **Basic Settings** et définir ses propriétés.

- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent.

- Si nécessaire, cliquez sur le bouton **Edit schema** pour voir les schémas d'entrée et de sortie, et apporter des modifications au schéma de sortie.

Le schéma de sortie de ce composant, comprend quatre colonnes standards de sortie en lecture seule.

- Dans le tableau **Key definition**, cliquez sur le bouton **[+]** pour ajouter à la liste les colonnes sur lesquelles vous souhaitez effectuer l'opération de mise en correspondance, *FirstName* et *LastName* dans ce scénario.
- Cliquez dans la première et la deuxième cellules de la colonne **Matching type** et sélectionnez dans la liste la (les) méthode(s) à utiliser pour l'opération de mise en correspondance, **Jaro-Winkler** dans cet exemple.
- Cliquez dans la première et la deuxième cellules de la colonne **Confidence Weight** et définissez les poids numériques pour chaque colonne utilisée comme attribut de clé.
- Cliquez sur le bouton **[+]** sous le tableau **Blocking Definition** afin d'ajouter une ligne, puis cliquez dans la ligne et sélectionnez dans la liste la colonne que vous souhaitez utiliser comme valeur de bloc, *Source* dans cet exemple.
  Utiliser une valeur de bloc réduit le nombre de paires d'enregistrements nécessitant examen. Les données d'entrée sont partitionnées en blocs exhaustifs basés sur la source des données. Cela réduit le nombre de paires à comparer, puisque la comparaison est limitée aux paires d'enregistrements dans chaque bloc.
- Double-cliquez sur le composant **tMap** afin d'ouvrir le **Map Editor**.

La zone d'entrée, à gauche, est déjà remplie par le schéma d'entrée provenant du composant précédent dans le Job.

- Cliquez sur le bouton **[+]** en haut à droite de la zone de sortie, afin d'ajouter autant de tables que nécessaire, deux dans cet exemple : *uniques* et *groups*. La première table regroupera les enregistrements de données uniques, et la deuxième regroupera tous les enregistrements ayant une distance de correpondance, dans l'enregistrement maître de chaque groupe.
- Déposez les colonnes d'entrée afin de remplir le premier schéma de sortie. Pour plus d'informations concernant le mapping des données, consultez *Mapping de flux de données* dans le **Guide Utilisateur Talend Open Studio**.
  Toutes les colonnes seront automatiquement remplies dans le **Schema Editor**, dans la moitié inférieure du **Map Editor**.
- Cliquez sur le bouton dans le coin supérieur droit de la première table de sortie afin d'ajouter une condition pour filtrer les données dans cette table : *row2.GRP_SIZE == 1*.
- Déposez les colonnes d'entrée pour remplir le deuxième schéma de sortie, et ajoutez le filtre suivant : *row2.GRP_SIZE > 1 || !row2.MASTER*.

- Dans le **Schema Editor** de la deuxième table de sortie, cliquez sur le bouton **[+]** afin d'ajouter deux colonnes supplémentaires : *weight* et *istarget*. La première mesurera la distance de correspondance et la seconde décidera si l'enregistrement est un enregistrement source ou cible.
- Cliquez sur **Ok** pour fermer le **Map Editor**.
- Dans l'espace de modélisation graphique, cliquez-droit sur le **tMap** et sélectionnez le lien *uniques*, et reliez-le au composant **tLogRow**. Connectez de la même manière le **tMap** au **tStewardshipTaskOutput** avec le lien *groups*.
- Double-cliquez sur le composant **tStewardshipTaskOutput** afin d'afficher sa vue **Basic settings** et définir ses propriétés.

- Dans la liste **Schema**, sélectionnez **Built-In** puis cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour ouvrir une boîte de dialogue.

Les données sont collectées des colonnes définies dans la table de sortie *groups* dans le composant **tMap**.

- Cliquez sur **OK** afin de fermer la boîte de dialogue et passer à l'étape suivante.
- Dans le champ **Host**, saisissez entre guillemets l'adresse IP du serveur de la base de données de la console de stewardship.
- Dans le champ **Port**, saisissez entre guillemets le numéro du port d'écoute du serveur de la base de données.
- Dans les champs **Username** et **Password**, saisissez votre identifiant et votre mot de passe de connexion au serveur MDM.
- Dans la liste **Type** list, sélectionnez le type des tâches que vous souhaitez écrire dans la console de stewardship : **Resolution** ou **Data**. Dans cet exemple, seules les tâches de résolution sont à écrire.
  Pour plus d'informations concernant les types de tâches, consultez le **Guide Talend Data Stewardship Console User Guide** (en anglais).
- Dans le champ **Created by**, saisissez entre guillemets le nom du créateur de la tâche, *Administrator* dans cet exemple. Le créateur de la tâche correspond à l'utilisateur de l'interface Web de **Talend MDM**. Pour plus d'informations, consultez le **Guide Talend MDM Web User Interface User Guide** (en anglais).
- Dans le champ **Owner**, saisissez entre guillemets le nom du propriétaire de la tâche, c'est-à-dire l'utilisateur auquel la tâche est assignée, *Administrator* dans cet exemple.

Une tâche peut être assignée à un utilisateur spécifique, soit à partir de la vue **Basic settings** du composant **tStewardshipTaskOutput**, soit directement à partir de la console de stewardship par un administrateur. Pour plus d'informations, consultez *tStewardshipTaskOutput, page 1428* et le **Guide Talend Data Stewardship Console User Guide** (en anglais).

- Dans le champ **Star**, saisissez entre guillemets le nombre d'étoiles, de 0 à 5, que vous souhaitez assigner à la tâche dans la console de stewardship afin de souligner son importance.
- Dans le champ **Tags**, saisissez entre guillemets le nom de la catégorie du tag associé aux tâches que vous souhaitez lire. Ce champ n'est pas utilisé dans ce scénario.
  Pour plus d'informations, consultez le **Guide Talend Data Stewardship Console User Guide** (en anglais).
- Dans la liste **Looping column**, sélectionnez une colonne du schéma d'entrée sur laquelle effectuer la boucle, *GID* dans cet exemple.
- Dans la liste **Source/Target selector**, sélectionnez la colonne qui décidera si l'enregistrement est un enregistrement source ou cible.
- Dans la liste **Source**, sélectionnez une colonne source dans le schéma d'entrée.
- Dans la liste **Score**, sélectionnez la colonne du score de correspondance dans le schéma d'entrée.
- Dans la liste **Weights**, sélectionnez la colonne définissant la distance de correspondance pour les colonnes d'entrée.
- Dans le tableau **Record Column**, cliquez sur le bouton **[+]** afin d'ajouter autant de lignes que vous souhaitez afficher dans chaque tâche dans **Talend Data Stewardship Console**. Cliquez dans chacune des lignes et sélectionnez la colonne que vous souhaitez afficher dans chaque tâche. Dans cet exemple, chaque tâche doit avoir quatre colonnes : *Firstname*, *Lastname*, *PostalCode* et *DOB*.
- Dans le tableau **Extra info**, cliquez sur le bouton **[+]** pour ajouter une ou plusieurs ligne(s) que vous pouvez utiliser afin d'ajouter des informations supplémentaires dans une ou plusieurs colonne(s) d'enregistrements de la tâche créée. Cliquez dans la ligne **Info** et sélectionnez une colonne d'entrée (ce qui ajoutera le nom de la colonne et sa valeur). Cliquez dans la ligne **Scope** puis saisissez le nom de la colonne d'enregistrements dans laquelle vous souhaitez ajouter les informations (nom de la colonne d'entrée et sa valeur).
- Double-cliquez sur le composant **tLogRow** afin d'afficher sa vue **Basic settings** et définir ses propriétés.
- Sauvegarder votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job Mr_GroupSampleWithTasks at 14:23 29/09/2010.

[statistics] connecting to socket on port 3816
[statistics] connected
.-------+-----------+--------+----------+----------+---+--------+------+-----.
|                                Uniques                                      |
|=------+-----------+--------+----------+----------+---+--------+------+----=|
|Source |Firstname  |Lastname|DOB       |PostalCode|GID|GRP_SIZE|MASTER|SCORE|
|=------+-----------+--------+----------+----------+---+--------+------+----=|
|Finance|Christopher|Town    |null      |78300     |1  |1       |true  |1.0  |
|POS    |Kristophe  |Toon    |01-01-1972|78450     |2  |1       |true  |1.0  |
|CRM    |Christophe |Toum    |31-03-1972|78540     |3  |1       |true  |1.0  |
|CRM    |Angelina   |Jolie   |null      |75007     |5  |1       |true  |1.0  |
'-------+-----------+--------+----------+----------+---+--------+------+-----'
[statistics] disconnected
Job Mr_GroupSampleWithTasks ended at 14:23 29/09/2010. [exit code=0]
```

La console de la vue **Run** affiche les quatre colonnes uniques du flux d'entrée.

L'identifiant pour chaque groupe (tâche) est affiché dans la colonne *GID* à côté de l'enregistrement correspondant. Le nombre d'enregistrements dans chacune des tâches est listé dans la colonne *GRP_SIZE* et calculé uniquement sur l'enregistrement maître. La colonne *MASTER* indique par la valeur **true** que l'enregistrement correspondant est un enregistrement maître. La colonne *SCORE* affiche la distance calculée entre l'enregistrement d'entrée et l'enregistrement maître selon l'algorithme de correspondance **Jargo-Winkler**.

Tous les autres enregistrements d'entrée ayant une distance de correspondance sont listés dans **Talend Data Stewardship Console**, en attendant qu'un "steward" fusionne, compare et collecte les enregistrements de données.

Composants XML

Ce chapitre passe en revue les propriétés des principaux composants présents dans la famille **XML** de la **Palette** de **Talend Open Studio**.

La famille **XML** regroupe les composants dédiés aux tâches liées au XML, notamment le parsing, la validation, la création de structures XML, etc.

tAdvancedFileOutputXML

Propriétés du tAdvancedFileOutputXML

Famille de composant	File/Output ou XML	
Fonction	Le composant **tAdvancedFileOutputXML** transmet des données vers un fichier de type XML et propose une interface de gestion des boucles et des Group by si nécessaire.	
Objectif	Le **tAdvancedFileOutputXML** crée un fichier XML contenant les valeurs des données définies dans l'arborescence XML.	
Basic settings	*File name*	Chemin d'accès et nom du fichier de sortie, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Configure XML tree	Ouvre l'interface d'aide à la création du fichier XML. Pour plus d'informations sur l'interface, consultez *Définir un arbre XML, page 1442*.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champ qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Sync columns	Cliquez sur ce bouton pour synchroniser le schéma de sortie avec celui d'entrée. La fonction Sync ne s'affiche que si une connexion de type Row est liée au composant de sortie.
	Append the source xml file	Cochez cette option pour ajouter de nouvelles lignes à la fin de votre fichier source XML.
	Generate compact file	Cochez cette case pour générer un fichier ne comprenant aucun espace vide ni aucun séparateur de ligne. L'ensemble des éléments se présente alors sur une ligne unique, ce qui permet de réduire le poids du fichier.

	Include DTD or XSL	Cochez cette case pour ajouter la déclaration DOCTYPE indiquant le nom de l'élément racine et le chemin d'accès et le nom du fichier DTD ou pour ajouter l'instruction de traitement indiquant le type de feuille de style (telle que les XSL) utilisée et son chemin d'accès et nom. **DTD** : saisissez le nom de l'élément racine dans le champ **Root Element** et saisissez le chemin d'accès et le nom du fichier DTD dans le champ **File Name**. **XSL** : saisissez le type de feuille de style dans le champ **Type** et saisissez le chemin d'accès et le nom du fichier dans le champ **File Name**.
Advanced settings	*Split output in several files*	Si le fichier XML de sortie est lourd, vous pouvez le scinder en plusieurs fichiers en définissant un certain nombre de lignes par fichier.
	Create directory only if not exists	Cette case est cochée par défaut. Cette option permet de créer le dossier contenant le fichier XML de sortie s'il n'existe pas déjà.
	Create empty element if needed	Cette case est cochée par défaut. Si le contenu de la colonne **Associated Column** est nul ou si aucune colonne n'est associée au noeud XML, cette option créera une balise ouvrante et une balise fermante aux endroits prévus.
	Create associated XSD file	Si l'un des éléments XML est associé à un espace de nommage, cette option créera le fichier XSD correspondant. Pour utiliser cette option, vous devez sélectionner le mode de génération **Dom4J** dans la liste **Generation mode**.
	Advanced separator (for number)	Cochez cette option pour modifier les séparateurs utilisés pour les nombres : **Thousands separator** : définissez le séparateur utilisé pour les milliers. **Decimal separator** : définissez le séparateur utilisé pour les décimaux.
	Generation mode	Sélectionnez le mode de génération le plus rapide mais consommant de la mémoire (**Fast but memory-consuming - Dom4J**) ou le plus lent ne consommant pas de mémoire (**Slow with no memory consumed**) en fonction de votre mémoire disponible.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	Don't generate empty file	Cochez cette case pour annuler la génération du fichier si celui-ci est vide.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log du Job, aussi bien au niveau du Job qu'au niveau de chaque composant.

Utilisation	Utilisez ce composant pour créer un fichier XML à partir des données transmises par d'autres composants via une connexion de type Row.
Limitation	n/a

Définir un arbre XML

Double-cliquez sur le composant **tAdvancedFileOutputXML** pour ouvrir l'interface dédiée ou cliquez sur le bouton **[...]** du champ **Configure Xml Tree** de l'onglet **Basic settings** dans la vue **Component**.

Le champ **Schema List** de la zone **Linker Source**, à gauche de l'interface, contient toutes les colonnes du flux d'entrée (à condition que le flux d'entrée soit connecté au composant **tAdvancedFileOutputXML**).

A droite de l'interface, dans la zone **Linker Target**, définissez la structure XML que vous souhaitez obtenir en sortie.

Vous pouvez la créer manuellement ou tout simplement importer la structure XML. Puis importez les colonnes du schéma d'entrée dans l'élément de l'arbre XML correspondant.

Importer un arbre XML

Le meilleur moyen de renseigner l'arbre XML est d'importer un fichier XML bien formé.

- Renommez la balise racine qui s'affiche par défaut dans le panneau **XML tree,** en cliquant sur celle-ci.
- Dans la colonne **XML Tree**, cliquez-droit sur le champ **root tag** pour afficher le menu contextuel.
- Dans le menu, sélectionnez **Import XML tree**.

- Sélectionnez le fichier à importer et cliquez sur **OK**.

Vous pouvez importer la structure XML d'un fichier au format XML, XSD et DTD.

XML Tree	Related Column	Node Status
VideoCollection		
Movie		
@id		-
Category		
Year		
Language		
Title		
Crew		
Director		
Cast		

La colonne **XML Tree** est donc automatiquement renseignée avec les éléments. Vous pouvez supprimer et ajouter des éléments ou des sous éléments à l'arbre :

- Sélectionnez l'élément adéquat dans l'arbre.
- Cliquez-droit pour faire apparaître le menu contextuel.
- Sélectionnez **Delete** pour supprimer la sélection de l'arbre ou sélectionnez l'option adéquate parmi les suivantes : **Add sub-element**, **Add attribute**, **Add namespace** pour enrichir l'arbre.

Créer manuellement l'arbre XML

Si vous ne possédez pas de structure XML déjà définie, vous pouvez la créer manuellement.

- Dans la colonne **XML Tree**, cliquez une fois sur le champ **root tag** pour le renommer.
- Cliquez droit sur ce champ pour afficher le menu contextuel.
- Dans le menu, sélectionnez **Add sub-element** pour créer le premier élément de la structure.

Vous pouvez aussi ajouter un attribut ou un élément enfant à n'importe quel élément de l'arbre ou supprimer n'importe quel élément de l'arbre.

- Sélectionnez l'élément adéquat dans l'arbre que vous venez de créer.
- Cliquez-droit à gauche du nom de l'élément pour afficher le menu contextuel.
- Dans le menu, sélectionnez l'option adéquate parmi les suivantes : **Add sub-element**, **Add attribute**, **Add namespace** ou **Delete**.

Mapping de données XML

Une fois votre arbre XML créé, vous pouvez alimenter chaque élément ou sous-élément XML avec les colonnes du flux d'entrée dans la colonne **Related Column** :

- Cliquez sur une des entrées de la colonne **Schema List**.
- Glissez-la dans le sous-élément correspondant à droite.
- Relâchez-la pour que le mapping soit effectif.

Une flèche bleue apparaît pour illustrer ce mapping. S'il est disponible, utilisez le bouton **Auto-Map** situé en bas à gauche de l'interface, il permet d'effectuer cette opération automatiquement.

Vous pouvez déconnecter n'importe quel mapping de n'importe quel élément de l'arbre XML :

- Sélectionnez l'élément de l'arbre XML que vous souhaitez déconnecter de sa source respective.
- Cliquez-droit à gauche de son nom pour afficher le menu contextuel.
- Sélectionnez **Disconnect linker**.

La flèche bleue disparaît.

Définir le statut du noeud

Définir l'arbre XML et le transfert de données ne suffit pas. Vous devez aussi définir l'élement sur lequel la boucle est effectuée et si nécessaire l'élément à partir duquel le regroupement est effectué.

Element Boucle

L'élément Boucle permet de définir l'élément objet d'une itération. L'élément Boucle est généralement un générateur de lignes.

Pour définir un élément comme un élément Boucle :

- Sélectionnez l'élément adéquat dans l'arbre XML.
- Cliquez-droit à gauche du nom de l'élément pour afficher le menu contextuel.
- Sélectionnez l'option **Set as Loop Element**.

La colonne **Node Status** affiche le nouveau statut sélectionné.

Il ne peut y avoir qu'un seul élément Boucle à la fois.

Elément Group

L'élément Group est optionnel, il représente un élément constant sur lequel est effectuée une fonction Groupby. L'élément Group ne peut être défini que si un élément Boucle a été préalablement défini.

Lorsque vous utilisez un élément Group, les lignes doivent être triées pour pouvoir être regroupées par le noeud sélectionné.

Pour définir un élément comme élément Group :

- Sélectionnez l'élément adéquat dans l'arbre XML.
- Cliquez-droit à gauche du nom de l'élément pour afficher le menu contextuel.
- Sélectionnez l'option **Set as Group Element**.

La colonne **Node Status** affiche le nouveau statut sélectionné et tout autre statut de regroupement est automatiquement défini, si nécessaire.

Une fois le mapping terminé, cliquez sur **OK** pour valider les paramètres et continuez la configuration de votre Job.

Scénario : Créer un fichier XML à l'aide d'une boucle

Le scénario suivant décrit la création d'un fichier XML à partir d'un fichier plat trié contenant une vidéothèque.

- Dans la palette de composants, cliquez sur les composants **tFileInputDelimited** et **tAdvancedFileOutputXML** et déposez-les dans l'espace de modélisation.
- Sinon, si vous avez déjà créé une métadonnée contenant la description du fichier d'entrée dans le noeud **Metadata** du **Repository**, alors vous pouvez cliquer-déposer directement cette métadonnée dans l'espace de modélisation pour paramétrer automatiquement le flux d'entrée.
- Cliquez-droit sur le composant d'entrée et reliez-le au composant **tAdvancedFileOutputXML** à l'aide d'une connexion de type Row Main.
- Sélectionnez le composant **tFileInputDelimited** et paramétrez ses propriétés dans la vue **Component** en bas de la fenêtre du Studio.

- Dans la liste déroulante **Property type**, sélectionnez l'option Repository si la description de votre fichier est stockée dans une métadonnée du Repository. Si vous cliquez-déposez le composant directement à partir de la métadonnée, vous n'aurez pas besoin de modifier ses propriétés.
- Sinon, sélectionnez l'option Built-in et renseignez manuellement les champs de l'onglet **Basic settings**.
- Le fichier d'entrée contient les colonnes suivantes, séparées par des points-virgules : *id*, *name*, *category*, *year*, *language*, *director* et *cast*.

- Dans cette exemple, le champ **Cast** regroupe différentes valeurs et **id** s'incrémente lorsque l'on change de film.
- Si nécessaire, définissez le schéma du **tFileDelimitedInput** en fonction de la structure du fichier.

- Une fois que vous avez vérifié que le schéma du fichier d'entrée correspond à vos attentes, cliquez sur **OK** pour valider.
- Puis sélectionnez le composant **tAdvancedFileOutputXML** et cliquez sur la vue **Component** pour paramétrer ses propriétés, ainsi que le mapping dans l'onglet **Basic settings**. Notez que vous pouvez directement double-cliquer sur le composant pour ouvrir l'interface de mapping.

- Dans le champ **File Name**, sélectionnez le fichier qui sera écrit à la fin du Job, s'il existe déjà ou saisissez le chemin d'accès et le nom du fichier qui seront créés automatiquement pour la sortie.
- Par défaut, le schéma (description du fichier) est automatiquement propagé à partir du flux d'entrée. Mais vous pouvez le modifier si nécessaire.
- Puis cliquez sur le bouton **[...]** ou double-cliquez sur le composant **tAdvancedFileOutputXML** dans l'espace de modélisation pour ouvrir l'éditeur de mapping.
- Les colonnes du schéma du fichier d'entrée sont listées à gauche de l'interface.

- A droite de l'interface, configurez l'arbre XML afin qu'il reflète la structure du fichier de sortie.
- Vous pouvez soit créer la structure noeud par noeud. Pour plus d'informations sur la création manuelle de l'arbre XML, consultez *Définir un arbre XML, page 1442*.
- Dans cet exemple, un modèle XML est utilisé pour renseigner automatiquement l'arbre XML.
- Cliquez-droit sur le champ **root tag** qui apparaît par défaut et sélectionnez l'option **Import XML tree** à la fin des options du menu contextuel.
- Dans la fenêtre **Ouvrir**, sélectionnez le fichier XML à importer et cliquez sur **OK** pour valider l'import.

Vous pouvez importer la structure XML d'un fichier au format XML, XSD et DTD.

- Puis cliquez-déposez chaque champ de la colonne **Schema List** vers son élément correspondant dans l'arbre XML de la colonne **XML tree** tel que décrit dans *Mapping de données XML, page 1443*.
- Le mapping apparaît sous la forme d'une flèche bleue entre la zone de gauche et celle de droite.

- Enfin, dans la colonne **Node Status**, définissez le statut du noeud sur lequel la boucle est effectuée. Dans cet exemple, l'élément *Cast* correspondant à l'élément Boucle puisque les données de ce champ dans le fichier d'entrée ne sont jamais les mêmes.
- Cliquez-droit sur l'élément Cast de l'arbre XML et sélectionnez l'option **Set as loop element**.
- Pour regrouper les données par film, l'élément Movie doit aussi être défini comme élément Group.
- Cliquez-droit sur le noeud parent Movie de l'arbre XML et sélectionnez l'option **Set as group element**.

- Le statut sélectionné paraît sur la ligne de l'élément correspondant dans la colonne **Node Status**.
- Cliquez sur **OK** pour valider la configuration.

Puis appuyez sur **F6** pour exécuter le Job.

VideoCollection2.xml

```
1  <?xml version="1.0" encoding="ISO-8859-15"?>
2
3  <VideoCollection>
4    <Movie id="1">
5      <Category>Comedy</Category>
6      <Year>1982</Year>
7      <Language>English</Language>
8      <Title>Tootsie</Title>
9      <Crew>
10       <Director>Sydney Pollack</Director>
11       <Cast>Dustin Hoffman</Cast>
12       <Cast>Jessica Lange</Cast>
13       <Cast>Sydney Pollack</Cast>
14     </Crew>
15   </Movie>
16   <Movie id="2">
17     <Category>Science Fiction</Category>
18     <Year>1997</Year>
19     <Language>French</Language>
20     <Title>The 5th Element</Title>
21     <Crew>
22       <Director>Eric Besson</Director>
23       <Cast>Bruce Willis</Cast>
```

Le fichier XML de sortie affiche la structure telle qu'elle a été définie.

tDTDValidator

Propriétés du tDTDValidator

Famille de composant	XML	
Fonction	Le composant **tDTDValidator** valide un fichier XML entrant en fonction d'une DTD et envoie les informations de validation à la sortie définie.	
Objectif	Ce composant permet de vérifier les données et la qualité de la structure du fichier traité	
Basic settings	*Schema type* et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Le schéma de ce composant est en lecture seule. Il contient les informations standard de validation de fichier.
	DTD file	Chemin d'accès au fichier DTD de référence.
	XML file	Chemin d'accès au fichier XML à valider.
	If XML is valid, display *If XML is not valid detected, display*	Saisissez un message à afficher dans la console **Run** basé sur le résultat de la comparaison.
	Print to console	Cochez cette case pour afficher le message de validation.
Utilisation	Ce composant peut être utilisé en standalone mais il est généralement connecté à un composant de sortie pour rassembler les informations de validation.	
Limitation	n/a	

Scénario : Valider un fichier XML

Ce scénario décrit un Job permettant de valider plusieurs fichiers d'un répertoire et de transmettre les informations de validation des fichiers non valides dans un fichier délimité.

- Cliquez et déposez les composants suivants de la **Palette** : **tFileList**, **tDTDValidator**, **tMap**, **tFileOutputDelimited**.
- Connectez le **tFileList** et le **tDTDValidator** avec un lien **Iterate** et les composants restants avec des liens Row **main**.

- Paramétrez les propriétés du composant tFileList afin qu'il récupère les fichiers XML du répertoire.

- Dans le champ **Filemask**, saisissez le masque **.xml* entre guillemets simples en Perl et guillemets doubles en Java.
- Paramétrez le chemin d'accès des fichiers XML à vérifier.
- Décochez la case **Case Sensitive**.
- Dans la vue **Component** du composant **tDTDValidate**, le schéma est en lecture seule puisqu'il contient des informations de validation standard.
- Paramétrez les propriétés du fichier DTD à utiliser comme référence, dans le champ **DTD file**.

- Dans le champ **Xml file**, appuyez sur **Ctrl+Espace** pour accéder à la liste des variables globales et sélectionnez la variable permettant de récupérer le chemin d'accès des fichiers : `$_globals{tFileList_1}{CURRENT_FILEPATH}` (en Perl)
- Dans les divers messages à afficher dans la console **Run**, utilisez JobName pour afficher le libellé du Job. Rappelez aussi le nom du fichier grâce à la variable : $_globals{tFileList_1}{CURRENT_FILE}. Saisissez votre message en tenant compte du langage utilisé pour le Job, Perl ou Java, notamment pour l'utilisation du point ou du plus.
- Cochez la case **Print to Console**.
- Dans le composant **tMap**, glissez-déposez les informations de validation du schéma standard que vous souhaitez passer au fichier de sortie.

- Une fois que le schéma de sortie est défini, ajoutez un filtre de condition pour ne sélectionner que les informations lorsque le fichier XML n'est pas valide.
- Saisissez d'abord la valeur souhaitée pour cette variable, l'opérateur en fonction du type de données filtrées puis la variable devant correspondre à cette condition, ici : `0 == $row1[validate]`
- Puis (si cela n'est pas déjà effectué) connectez le **tMap** au **tFileOutputDelimited** via un lien main row. Nommez-le *errorsOnly*.
- Dans l'onglet **Basic settings** du composant **tFileOutputDelimited**, paramétrez le chemin d'accès de destination du fichier délimité, les séparateurs de champs et l'encodage.
- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

```
Starting job tDTDValidator_oj at 14:28 14/10/2010.

[statistics] connecting to socket on port 3420
[statistics] connected
[job tDTDValidator_oj]employee_1.xml is Valid
[job tDTDValidator_oj]employee_2.xml is Invalid
[job tDTDValidator_oj]employee_3.xml is Valid
[job tDTDValidator_oj]employee_4.xml is Invalid
[job tDTDValidator_oj]employee_5.xml is Invalid
[statistics] disconnected
Job tDTDValidator_oj ended at 14:28 14/10/2010. [exit
code=0]
```

Dans la console de la vue **Run**, les messages définis apparaissent pour chaque fichier invalide. Au même moment, le fichier de sortie est alimenté avec les informations de validation.

tExtractXMLField

Propriétés du tExtractXMLField

Famille de composant	XML	
Fonction	Le composant **tExtractXMLField** lit la structure XML contenue dans les champs d'un fichier ou d'une table et en extrait les données.	
Objectif	Le **tExtractXMLField** lit la structure XML contenue dans les champs d'un fichier ou d'une table, récupère les données contenues dans cette structure et les envoie au composant suivant via un lien Row.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement..
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	XML field	Nom du champ XML à traiter. Voir également : *Définir les variables à partir de la vue Component* du **Guide utilisateur** de **Talend Open Studio**
	Loop XPath query	Noeud de l'arborescence XML sur lequel est basé la boucle.
	Mapping	**Column** : reflète le schéma tel qu'il est défini dans l'éditeur Edit Schema. **XPath Query** : Saisissez les champs à extraire de la structure Xml d'entrée. **Get nodes** : Cochez cette case pour récupérer le contenu XML de tous les nœuds courants spécifiés dans le champ **Xpath Query**, ou cochez la case correspondant à un nœud spécifique pour ne récupérer que le contenu du nœud sélectionné.
	Limit	Nombre maximum de lignes à traiter. Si Limit = 0, aucune ligne n'est lue ou traitée.

	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log du Job, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Ce composant doit être utilisé en composant intermédiaire. Il nécessite donc un composant d'entrée et de sortie. Il permet d'extraire des données d'un champ délimité à l'aide d'un lien **Row** > **Main**, et permet de créer un flux de rejet avec un lien **Row** > **Reject**, filtrant les données dont le type ne correspond pas au type défini.	
Limitation	n/a	

Scénario 1 : Extraire les données XML d'un champ d'une table de base de données

Ce scénario est composé de trois composants et permet de lire la structure XML contenue dans les champs d'une base de données et d'en extraire les données.

- Glissez les composants **tMysqlInput** de la famille **Database** > **Mysql**, **tExtractXMLField** de la famille **Xml** et un composant **tFileOutputDelimited** de la famille **File** > **Output** de la Palette dans le Job designer.
- Reliez les composants entre eux via des connexions de type **Main** > **Row**.
- Double-cliquez sur le composant **tMysqlInput** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component** :

- Si votre schéma est déjà stocké sous le noeud **Db Connections** dans le **Repository**, sélectionnez l'option **Repository** dans le champ **Schema Type** puis choisissez les métadonnées appropriées à partir de la liste.
- Si vous n'avez encore défini aucun schéma, sélectionnez l'option **Built-in** et renseignez manuellement les informations de connexion et la structure des données dans un schéma. Pour plus d'informations sur les propriétés du composant **tMysqlInput**, consultez *tMysqlInput, page 476.*
- Dans le champ **Table Name**, saisissez le nom de la table dont les champs contiennent les données XML. Dans ce scénario, la table se nomme *customerdetails* et ne contient qu'une seule colonne.
- Cliquez sur le bouton **Guess Query** pour récupérer la requête correspondant à votre schéma.
- Dans le Job designer, double-cliquez sur le composant **tExtractXMLField** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component** :

- Dans la liste déroulante **Property type**, sélectionnez l'option **Repository** si la description de votre fichier est stockée dans une métadonnée du Repository. Si vous cliquez-déposez le composant directement à partir de la métadonnée, vous n'aurez pas besoin de modifier ses propriétés.
- Sinon, sélectionnez l'option **Built-in** et renseignez manuellement les champs suivants.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent. Vous pouvez cliquer sur le bouton **[...]** à côté du champ **Edit schema** pour consulter le schéma.
- Dans la liste **Xml field**, sélectionnez la colonne dont vous souhaitez extraire le contenu XML. La colonne se nomme *CustomerDetails* et est l'unique colonne de la table *customerdetails*.
- Dans le champ **Loop XPath query**, saisissez le noeud de l'arborescence sur lequel effectuer la boucle afin de récupérer les données.
- La colonne **Column** du tableau **Mapping** sera automatiquement renseignée avec la description du fichier que vous avez configuré. Dans la colonne **Xpath query**, saisissez entre guillemet le noeud du champ XML contenant les données que vous voulez extraire. Dans ce scénario, l'objectif est d'extraire les noms des clients, saisissez donc *"CustomerName"* qui correspond à l'élément XML contenant cette information.
- Dans le Job designer, double-cliquez sur le composant **tFileOutputDelimited** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component** :

- Dans le champ **File Name**, indiquez le répertoire et le nom du fichier de sortie. Pour ce scénario, appelez le fichier *CustomerNames.csv*.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent. Vous pouvez cliquer sur le bouton **[...]** à côté du champ **Edit schema** pour consulter le schéma.
- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

```
1   Griffith Paving and Sealcoatin
2   Bill's Dive Shop
3   Childress Child Day Care
4   Facelift Kitchen and Bath
5   Terrinni & Son Auto and Truck
6   Kermit the Pet Shop
7   Tub's Furniture Store
8   Toggle & Myerson Ltd
9   Childress Child Day Care
10  Elle Hypnosis and Therapy Cent
11  Lennox Air Pollution Control
```

Le Job retourne les noms des clients contenus sous le noeud `CustomerName` du champ *CustomerDetails* de la table.

Scénario 2 : Extraire les données valides et les données erronées à partir d'un champ XML dans fichier délimité

Ce scénario est composé de quatre composants et permet de lire la structure XML contenue dans les champs d'un fichier délimité, d'en extraire les données et de rejeter les données erronées.

- A partir de la **Palette**, glissez dans le Job designer les composants **tFileInputDelimited** de la famille **File** > **Input**, **tExtractXMLField** de la famille **Xml**, **tFileOutputDelimited** de la famille **File** > **Output** et **tLogRow** de la famille **Logs & Errors**.
- Reliez les composants entre eux comme indiqué sur le schéma ci-dessus.
- Double-cliquez sur le composant **tFileInputDelimited** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component** :

- Sélectionnez l'option **Built-in** à partir de la liste déroulante **Schema** et renseignez manuellement les informations concernant le fichier délimité dans les champs correspondants.
- Cliquez sur le bouton **[...]** à côté du champ **Edit schema** pour afficher la boîte de dialogue permettant de définir la structure des données, comme présenté ci-dessus.
- Cliquez sur le bouton **[+]** pour ajouter autant de lignes que nécessaire à la structure des données. Dans cet exemple, il n'y a qu'une seule colonne dans le schéma : *xmlStr*.
- Cliquez sur **OK** pour valider vos changements et fermer la boîte de dialogue.

Si vous avez préalablement stocké votre schéma dans le répertoire **Metadata** sous **File delimited**, sélectionnez l'option **Repository** dans la liste déroulante **Schema** puis cliquez sur le bouton [...] à côté du champ pour afficher la boîte de dialogue **[Repository Content]** à paritr de laquelle vous pouvez sélectionner un schéma pertinent. Cliquez sur **Ok** pour fermer la boîte de dialogue et remplir les champs automatiquement avec les métadonnées contenues dans le schéma.
Pour plus d'informations sur le stockage des schémas de métadonnées dans le **Repository**, consultez les sections *Configurer un schéma File Delimited* et *Déposer des composants à partir du répertoire Metadata* du **Guide Utilisateur** de **Talend Open Studio**.

- A partir de la liste **Property type**, sélectionnez :
  -**Repository** si vous avez déjà stocké les métadonnées de votre fichier d'entrée dans le **Repository**, les champs suivants sont alors renseignés automatiquement avec les informations stockées, ou
  -**Built-in** si aucune métadonnée n'est stockée pour votre fichier d'entrée ; vous devez alors renseigner manuellement les champs suivants.
  Dans cet exemple, c'est le mode **Built-in** qui est utilisé.
- Dans le champ **File Name**, renseignez le chemin d'accès ou cliquez sur le bouton **[...]** pour parcourir vos dossiers jusqu'au fichier à traiter. Dans cet exemple, il s'agit du fichier *CustomerDetails_Error.*
  Ce fichier csv stocké localement contient une suite de lignes simples au format XML, séparées par un double retour chariot.

- Dans le champ **Row Separator**, renseignez le séparateur de lignes, ici le double retour chariot. Dans le champ **Field Separator**, renseignez le séparateur de champ, aucun ici.
- Dans cet exemple, il n'y a pas d'en-tête (**Header**), de pied de page (**Footer**) ou de nombre limite de lignes à traiter (**Limit**).
- Dans l'espace graphique, double-cliquez sur le composant **tExtractXMLField** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component** :

- A partir de la liste **Property type**, sélectionnez :
  -**Repository** si vous avez déjà stocké les métadonnées de votre fichier d'entrée dans le **Repository**, les champs suivants sont alors renseignés automatiquement avec les informations stockées, ou
  -**Built-in** si aucune métadonnée n'est stockée pour votre fichier d'entrée ; vous devez alors renseigner manuellement les champs suivants.
  Dans cet exemple, c'est le mode **Built-in** qui est utilisé.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent. Vous pouvez cliquer sur le bouton **[...]** à côté du champ **Edit schema** pour consulter le schéma ou le modifier.
  La colonne **Column** du tableau **Mapping** sera automatiquement renseignée avec lae schéma défini.
- Dans la liste **Xml field**, sélectionnez la colonne dont vous souhaitez extraire le contenu XML. Dans cet exemple, la colonne se nomme *xmlStr*.
- Dans le champ **Loop XPath query**, saisissez le noeud de l'arborescence sur lequel effectuer la boucle afin de récupérer les données.
- Dans la colonne **Xpath query**, saisissez entre guillemets le noeud du champ XML contenant les données que vous voulez extraire.
- Dans le Job designer, double-cliquez sur le composant **tFileOutputDelimited** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component** :

- Choisissez le mode **Built-in** dans la liste déroulante **Property Type**.
- Dans le champ **File Name**, indiquez le répertoire et le nom du fichier de sortie. Pour ce scénario, appelez le fichier *CustomerNames_right.csv*.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent. Vous pouvez cliquer sur le bouton **[...]** à côté du champ **Edit schema** pour consulter le schéma ou le modifier.
- Dans le Job designer, double-cliquez sur le composant **tLogRow** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component**.
- Cliquez sur le bouton **Sync Columns** pour récupérer le schéma du composant précédent. Pour plus d'informations sur le composant **tLogRow**, consultez le chapitre *tLogRow, page 1163*.
- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

```
Starting job A at 14:22 05/11/2009.

[statistics] connecting to socket on port 3845
[statistics] connected
|
<CustomerDetails>
  <CustomerName>Childress Child Day Care</Customername>
  <id>9</id>
</CustomerDetails>||Error on line 3 of document  : The element type
"CustomerName" must be terminated by the matching end-tag "</CustomerName>".
Nested exception: The element type "CustomerName" must be terminated by the
matching end-tag "</CustomerName>". - Line: 8
[statistics] disconnected
Job A ended at 14:22 05/11/2009. [exit code=0]
```

Le composant **tExtractXMLField** lit et extrait dans un fichier délimité en sortie, *CustomerNames_right* dans cet exemple, les informations concernant les clients dont la structure XML est correcte, et affiche les lignes erronées dans la console de log.

tFileInputXML

Propriétés du tFileInputXML

Famille du composant	File/Input	
Fonction	Le composant **tFileInputXML** lit un fichier XML structuré ou un flux de données et extrait les données ligne par ligne.	
Objectif	Ce composant lit un fichier XML structuré ou un flux de données ligne par ligne pour le scinder en champs et envoie les champs tel que définis dans le schéma au composant suivant du Job via une connexion **Row**.	
Basic settings	*Property type*	Peut être Built-in ou Repository.
		Built-in : Propriétés utilisées ponctuellement.
		Repository : Sélectionnez le fichier de propriétés du composant. Les champs suivants sont alors pré-remplis à l'aide des données collectées.
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	File name/Stream	**File name** : Chemin d'accès et nom du fichier, et/ou variable à traiter. **Stream** : Flux de données à traiter. Les données doivent préalablement être mises en flux afin d'être récupérées par le **tFileInputXML** via la variable *INPUT_STREAM* disponible dans la liste d'auto-complétion (**Ctrl+Espace**). Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Loop XPath query	Noeud de l'arborescence sur lequel la boucle est basée.

	Mapping	**Column** : Colonnes à mapper. Elles reflètent le schéma défini par le champ **Schema type**. **XPath Query** : Saisissez les champs à extraire de l'entrée structurée. **Get nodes** : Cochez cette case pour récupérer le contenu XML de tous les nœuds courants spécifiés dans le champ **Xpath Query**, ou cochez la case correspondant à un nœud spécifique pour ne récupérer que le contenu du nœud sélectionné. L'option **Get Nodes** fonctionne en mode **DOM4j** et **SAX**, cependant avec le mode **SAX**, les noms d'espace ne sont pas supportés. Pour plus d'informations concernant les modes **DOM4j** et **SAX**, consultez les propriétés de la liste **Generation mode** de l'onglet **Advanced settings**.
	Limit	Nombre maximum de lignes à traiter. Si Limit = 0, aucune ligne n'est lue ou traitée. Si -1, toutes les lignes sont lues ou traitées, qiuelque soit le nombre.
	Die on error	Cette case est cochée par défaut et stoppe le Job en cas d'erreur. Décochez cette case pour terminer le traitement avec les lignes sans erreurs, et ignorer les lignes en erreur. Vous pouvez récupérer les lignes en erreur, si vous le souhaitez. Pour cela, utilisez un lien **Row** > **Reject**.
Advanced settings	*Advanced separator (for number)*	Cochez cette option pour modifier les séparateurs utilisés pour les nombres : **Thousands separator** : définissez le séparateur utilisé pour les milliers. **Decimal separator** : définissez le séparateur utilisé pour les décimaux.
	Ignore the namespaces	Cochez cette case pour ignorer les espaces de noms. **Generate a temporary file** : renseignez le chemin d'accès et le nom du fichier temporaire XML.
	Use Separator for mode Xerces	Cochez cette case pour séparer les valeurs des noeuds fils concaténées. Ce champ ne peut être utilisé que si vous avez sélectionné le mode **Xerces** dans le champ **Generation mode**. Le champs **Field separator** apparaît : Définissez le séparateur à utiliser pour séparer les valeurs
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.

	Generation mode	Sélectionnez dans la liste déroulante le mode de génération du fichier XML en fonction de la disponibilité de votre mémoire et de la rapidité souhaitée. -**Fast and memory-consuming (Dom4j)** : plus rapide mais consommateur de mémoire, -**Memory-consuming (Xerces)** : consommateur de mémoire, -**Less memory consumed (SAX)** :moins consommateur de mémoire. Lorsque que vous utilisez l'option **Get Nodes** du tableau **Mapping** de l'onglet **Basic settings** avec le mode de génération **SAX**, les noms d'espace ne sont pas supportés.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log du Job, aussi bien au niveau du Job qu'au niveau de chaque composant.
Utilisation	Ce composant doit être utilisé en tant que composant d'entrée. Il permet de créer un flux de données XML à l'aide d'un lien **Row** > **Main**, ainsi que de créer un flux de rejet avec un lien **Row** > **Reject** filtrant les données dont le type ne correspond pas au type défini. Pour un exemple d'utilisation de ces deux liens, consultez le *Scénario 2 : Extraire les données XML erronées dans un flux de rejet, page 1466*.	
Limitation	n/a	

Scénario 1 : Extraire des adresses XML

Ce scénario très basique est constitué de deux composants : un composant **tFileInputXML** qui extrait des données du fichier XML contenant des adresses et un composant **tLogRow** qui affiche le résultat de cette opération dans la console **Run** via le composant **tLogRow**.

- Sélectionnez un composant **tFileInputXML** dans le dossier **File** de la **Palette**. Cliquez et déposez également un composant **tLogRow** et connectez ces deux composants.
- Dans l'onglet **Basic settings** du composant **tFileInputXML**, définissez les propriétés :

- Le fichier XML contenant les adresses a été préalablement défini dans la zone Metadata, par conséquent récupérez ses propriétés en sélectionnant **Repository** dans **Property type**. De cette manière, les propriétés sont automatiquement récupérées et le reste des champs est renseigné (à l'exception du Schéma). Pour plus d'informations concernant l'assistant de création de métadonnées, consultez du **Guide utilisateur** de **Talend Open Studio**.
- De la même manière, sélectionnez le schéma approprié dans la liste de métadonnées du **Repository**. Cliquez sur **Edit schema** si vous souhaitez modifier le schéma chargé.
- Dans le champ **Filename**, indiquez le fichier structuré d'entrée.
- Dans le champ **Loop XPath query**, modifiez le noeud de boucle si nécessaire.
- Dans la table de **Mapping**, renseignez les champs à extraire et à afficher dans la sortie.
- Si le fichier est volumineux, définissez un nombre de ligne maximum à traiter dans le champ **Limit**.
- Saisissez l'encodage si nécessaire, puis double-cliquez sur tLogRow pour définir le caractère séparateur.
- Enfin, appuyez sur **F6** ou cliquez sur le bouton **Run** dans la vue **Run** pour exécuter le Job. Dans la console, les champs définis sont extraits du fichier XML structuré et sont affichés.

```
Starting job XMLStreetFinder at 12:42 05/01/2007.
Paris|2eme arrondissement|Rue de la Paix
Paris|8eme arrondissement|Champs Elysees
New York City|Manhattan|Madison avenue
New York City|Brooklyn|Washington heights
Job XMLStreetFinder ended at 12:42 05/01/2007. [exit code
```

Scénario 2 : Extraire les données XML erronées dans un flux de rejet

Ce scénario décrit un Job à trois composants qui permet de lire un fichier XML et :

- d'une part, de retourner les données XML correctes dans un autre fichier XML,
- et d'autre part, d'afficher dans la console de log les données erronées dont le type ne correspond pas au type défini.

- A partir de la **Palette**, glissez les composants **tFileInputXML**, **tFileOutputXML** et **tLogRow** dans l'espace de modélisation.
- Cliquez-droit sur le composant **tFileInputXML**, sélectionnez **Row** > **Main** dans le menu contextuel, et cliquez sur le composant **tFileOutputXML** pour créer le lien.
- Cliquez-droit sur le composant **tFileInputXML**, sélectionnez **Row** > **Reject** dans le menu contextuel, et cliquez sur le composant **tLogRow** pour créer le lien de rejet.
- Double-cliquez sur le composant **tFileInputXML** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés.

- Dans la liste **Property Type**, sélectionnez **Repository** et cliquez sur le bouton **[...]** pour afficher la boîte de dialogue **[Repository Content]** à partir de laquelle vous pouvez sélectionner la métadonnée relative à votre fichier si vous l'avez stocké sous le noeud **File xml** du répertoire **Metadata** du **Repository**. Les champs suivants seront renseignés automatiquement. Sinon, sélectionnez le mode **Built-in** et renseignez ces champs manuellement.
  Pour plus d'informations sur le stockage des schémas de métadonnées dans le **Repository**, consultez les sections *Configurer un Schéma File XML* et *Déposer des composants à partir du répertoire Metadata* du **Guide Utilisateur** de **Talend Open Studio**.

- Dans le champ **Schema Type**, sélectionnez **Repository** et cliquez sur le bouton **[...]** pour sélectionner le schéma décrivant la structure de votre fichier si vous l'avez stocké dans le **Repository**. Sinon, cliquez sur le bouton **[...]** à côté du champ **Edit schema** et définissez le schéma manuellement.

Dans cet exemple, le schéma comprend cinq colonnes : *id*, *CustomerName*, *CustomerAddress*, *idState* et *id2*.

- Cliquez sur le bouton [...] à côté du champ **Filename** et indiquez le chemin d'accès au fichier XML.

- Dans le champ **Loop XPath query**, saisissez, entre guillemets, le chemin du noeud sur lequel effectuer une boucle.

Dans la table **Mapping**, la colonne **Column** est automatiquement renseignée avec les colonnes définies dans le schéma.

- Dans la colonne **XPath query**, saisissez entre guillemets le noeud du fichier XML contenant les données que vous voulez extraire dans la colonne correspondante.

- Dans le champ **Limit**, saisissez le nombre de lignes à traiter. Dans ce scénario, l'objectif est de traiter les 10 premières lignes.

- Double-cliquez sur le composant **tFileOutputXML** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés.

- Cliquez sur le bouton **[...]** à côté du champ **File Name** pour sélectionner le répertoire de destination du fichier et saisissez le nom du fichier. Dans ce scénario, appelez le fichier *customer_data.xml*.
- Dans le champ **Row tag**, saisissez entre guillemets le nom que vous souhaitez donner à la balise contenant les données.
- Cliquez sur **Edit schema** et assurez-vous que le schéma est récupéré du schéma d'entrée. Si nécessaire, cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
- Double-cliquez sur le composant **tLogRow** pour afficher l'onglet **Basic settings** de la vue **Component** et paramétrer ses propriétés.
- Cliquez sur **Edit schema** et assurez-vous que le schéma est récupéré du schéma d'entrée. Si nécessaire, cliquez sur **Sync columns** pour récupérer le schéma du composant précédent.
- Dans la zone **Mode**, sélectionnez l'option **Vertical**.
- Enregistrez votre Job et appuyez sur **F6** pour l'exécuter.

```
Starting job tFileInputXML at 14:34 06/11/2009.

[statistics] connecting to socket on port 3996
[statistics] connected
.-------------------------------------------------------.
|                    #1. tLogRow_1                      |
+-----------------+-----------------------------------+
| key             | value                             |
+-----------------+-----------------------------------+
| id              | null                              |
| CustomerName    | null                              |
| CustomerAddress | null                              |
| idState         | null                              |
| id2             | null                              |
| errorCode       | null                              |
| errorMessage    | For input string: "ab" - Line: 1  |
+-----------------+-----------------------------------+
.-------------------------------------------------------.
|                    #2. tLogRow_1                      |
+-----------------+-----------------------------------+
| key             | value                             |
+-----------------+-----------------------------------+
| id              | 9                                 |
| CustomerName    | Childress Child Day Care          |
| CustomerAddress | 788 Tennyson Ave.                 |
| idState         | 12                                |
| id2             | null                              |
| errorCode       | null                              |
| errorMessage    | For input string: "cd" - Line: 8  |
+-----------------+-----------------------------------+
[statistics] disconnected
Job tFileInputXML ended at 14:35 06/11/2009. [exit code=0]
```

Le fichier *customer_data.xml* contenant les données correctes est créé et les données erronées sont affichées dans la console de la vue **Run**.

tFileOutputXML

Propriétés de tFileOutputXML

Famille de composant	File/Output	
Fonction	Le composant **tFileOutputXML** produit en sortie des données sous forme de fichier de type XML.	
Objectif	Le **tFileOutputXML** écrit dans un fichier XML des données séparées selon un schéma spécifié.	
Basic settings	*File name*	Chemin d'accès et nom du fichier de sortie, et/ou variable à utiliser. Pour plus d'informations concernant l'utilisation et la définition de variables, consultez la section *Définir les variables à partir de la vue Contexts* du **Guide utilisateur** de **Talend Open Studio**.
	Root tag	Enveloppe les données et la structure entière du fichier de sortie.
	Row tag	Enveloppe les données et la structure de chaque ligne.
	Column name as tag name	Cochez cette case pour récupérer les libellés de colonnes du schéma d'entrée et les utiliser comme balise XML.
	Split output in files	Si le fichier XML en sortie et volumineux, vous pouvez scinder ce fichier en plusieurs parties.
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champ qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Sync columns	Cliquez sur ce bouton pour synchroniser le schéma de sortie avec le schéma d'entrée. La fonction Sync ne s'affiche que si une connexion de type Row est liée au composant de sortie.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez Custom et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.

Utilisation	Utilisez ce composant pour écrire dans un fichier XML des données récupérées d'autres composants à l'aide d'une connexion de type Row.
Limitation	n/a

Scénario : Utiliser un format de sortie XML

Un scénario utilisant le composant tFileOutputXML est décrit dans le *Scénario : Transformer un fichier positionnel en XML, page 958*.

tWriteXMLField

Propriétés du tWriteXMLField

Famille de composant	XML	
Fonction	Le composant **tWriteXMLField** lit un fichier XML d'entrée, en extrait la structure et l'insère dans les champs du fichier ou de la table de sortie.	
Objectif	Le **tWriteXMLField** lit un fichier XML d'entrée, extrait la structure du fichier et l'insère dans les champs du fichier ou de la table de sortie.	
Basic settings	*Output Column*	Sélectionnez la colonne du composant de sortie dans laquelle vous souhaitez écrire la structure XML.
	Configure Xml Tree	Ouvre l'interface d'aide à la création de la structure XML à écrire dans un champ. Pour plus d'informations sur l'interface, consultez *Définir un arbre XML, page 1442*.
	Schema type et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository.
		Built-in : Le schéma est créé et conservé ponctuellement pour ce composant seulement. Voir également : *Paramétrer un schéma built-in* du **Guide utilisateur** de **Talend Open Studio**.
		Repository : Le schéma existe déjà et est stocké dans le Repository. Ainsi, il peut être réutilisé dans divers projets et Job designs. Voir également : *Paramétrer un schéma du Repository* du **Guide utilisateur** de **Talend Open Studio**.
	Sync columns	Cliquez sur ce bouton pour synchroniser le schéma de sortie avec celui d'entrée. La fonction Sync ne s'affiche que si une connexion de type Row est liée au composant de sortie.
	Group by	Sélectionnez la colonne à utiliser pour regrouper les données.
Advanced settings	*Remove the xml declaration*	Cochez cette case si vous ne souhaitez pas inclure la déclaration XML.
	Create empty element if needed	Cette case est cochée par défaut. Si le contenu de la colonne **Related Column** de l'interface permettant de créer la structure XML est nul, ou si aucune colonne n'est associée au noeud XML, cette option créera une balise ouvrante et une balise fermante aux endroits prévus.

	Create associated XSD file	Si l'un des éléments XML est associé à un espace de nommage, cette option créera le fichier XSD correspondant. Pour utiliser cette option, vous devez sélectionner le mode de génération **Dom4J** dans la liste **Generation mode**.
	Advanced separator (for number)	Cochez cette case pour modifier les séparateurs utilisés par défaut dans les nombres. **Thousands separator** : saisissez entre guillemets le séparateur à utiliser pour les milliers. **Decimal separator** : saisissez entre guillemets le séparateur à utiliser pour les décimales.
	Generation mode	Sélectionnez le mode de génération le plus rapide mais consommant de la mémoire (**Fast but memory-consuming - Dom4J**) ou le plus lent ne consommant pas de mémoire (**Slow with no memory consumed**) en fonction de votre mémoire disponible.
	Encoding	Sélectionnez l'encodage à partir de la liste ou sélectionnez **Custom** et définissez-le manuellement. Ce champ est obligatoire pour la manipulation des données de base de données.
	tStatCatcher Statistics	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant doit être utilisé en composant intermédiaire. Il nécessite donc un composant d'entrée et de sortie.	
Limitation	n/a	

Scénario : Extraire la structure d'un fichier XML et l'insérer dans les champs d'une base de données

Ce scénario est composé de trois composants et permet de lire un fichier XML, d'en extraire la structure puis de transmettre cette structure dans les champs d'une table.

- Glissez les composants **tFileInputXml** et **tWriteXMLField** de la famille **Xml** et un composant **tMysqlOutput** de la famille **Database** > **Mysql** de la Palette dans le Job designer.
- Reliez ces composants via des liens de type **Row** > **Main**.
- Double-cliquez sur le composant **tFileInputXml** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component** :

- Dans la liste déroulante **Property type**, sélectionnez l'option **Repository** si la description de votre fichier est stockée dans une métadonnée du Repository. Si vous cliquez-déposez le composant directement à partir de la métadonnée, vous n'aurez pas besoin de modifier ses propriétés.
- Sinon, sélectionnez l'option **Built-in** et renseignez manuellement les champs suivants. Pour plus d'informations sur les propriétés du composant **tFileInputXML**, consultez *tFileInputXML, page 1462*.
- Si vous avez sélectionnez l'option **Built-in**, cliquez sur le bouton [...] à côté du champ **Edit schema** puis dans la boîte de dialogue, décrivez manuellement la structure de votre fichier.
- Dans le champ **Look Xpath query**, saisissez sur quel noeud du fichier la boucle doit être effectuée. Pour ce scénario, la boucle est effectuée sur le noeud customer.
- La colonne **Column** du tableau **Mapping** sera automatiquement renseignée avec la description du fichier que vous avez configuré. Dans la colonne **Xpath query**, saisissez entre guillemet le noeud du fichier XML contenant les données correspondant à la colonne de gauche.
- Cliquez sur le composant **tWriteXMLField** dans le Job designer puis cliquez sur la vue **Component** pour l'afficher et paramétrer les propriétés du composant :

- Cliquez sur le bouton [...] à côté du champ **Edit schema** et, dans la boîte de dialogue, ajoutez une ligne en cliquant sur le bouton (+).

- Dans cette ligne, saisissez le nom de la colonne de votre flux de sortie que vous souhaitez renseigner avec la structure du fichier XML dans la zone de droite, *CustomerDetails* dans ce scénario.
- Dans la colonne **Type** de cette ligne, indiquez qu'elle est de type *String* et dans la colonne **Length**, indiquez qu'elle est de longueur *255*.
- Cliquez sur **Ok** pour valider votre schéma de sortie et retourner aux propriétés du composant.
- Dans le champ **Output Column**, sélectionnez la colonne dans laquelle vous souhaitez insérer le contenu XML.
- Cliquez sur le bouton [...] du champ **Configure Xml Tree** pour ouvrir l'interface d'aide à la création de structures XML.

- Dans la zone **Link Target**, cliquez sur le noeud *rootTag* et renommez-le *CustomerDetails*.
- Dans la zone **Linker source**, sélectionnez *CustomerName* et *CustomerAddress* et glissez-les sur le noeud *CustomerDetails*. Une boîte de dialogue s'ouvre vous demandant quel type d'action vous souhaitez effectuer : sélectionnez **Create as sub-element of target node** afin de créer des sous-éléments au noeud *CustomerDetails*.
- Cliquez-droit sur l'élément *CustomerName* et sélectionnez l'option **Set As Loop Element** dans le menu.
- Cliquez sur **OK**.

- Dans le Job designer, double-cliquez sur le composant **tMysqlOutput** pour paramétrer ses propriétés dans l'onglet **Basic settings** de la vue **Component** :

- Si votre schéma est déjà stocké sous le noeud **Db Connections** dans le **Repository**, sélectionnez l'option **Repository** dans le champ **Schema Type** puis choisissez les métadonnées appropriées à partir de la liste.
- Si vous n'avez encore défini aucun schéma, sélectionnez l'option **Built-in** et renseignez manuellement les informations de connexion et la structure des données dans un schéma. Pour plus d'informations sur les propriétés du composant **tMysqlOutput**, consultez *tMysqlOutput, page 489.*
- Dans le champ **Table**, saisissez le nom de la table dont les champs contiendront les données XML.
- Dans le champ **Action on table**, sélectionnez l'opération que vous souhaitez effectuer sur la table. Pour ce scénario, sélectionnez **Create table** pour créer la table.
- Dans le champ **Action on data**, sélectionnez l'opération que vous souhaitez effectuer sur les données. Pour ce scénario, laissez l'option **Insert**.
- Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent. Vous pouvez cliquer sur le bouton [...] à côté du champ **Edit schema** pour consulter le schéma.
- Enregistrez le Job et appuyez sur **F6** pour l'exécuter.

Chaque champ de la colonne **CustomerDetails** est renseignée avec la structure XML du fichier de départ : l'instruction de traitement XML `<?xml version=""1.0"" encoding=""ISO-8859-15""?>`, le premier noeud séparant chaque client `<CustomerDetails>` et les informations sur les clients `<CustomerAddress>` et `<CustomerName>`.

tXSDValidator

Propriétés du tXSDValidator

Famille de composant	XML	
Fonction	Le composant **tXSDValidator** valide un fichier XML entrant en fonction d'un fichier XSD et envoie les informations de validation à la sortie définie.	
Objectif	Ce composant permet de vérifier les données et la qualité de la structure du fichier traité	
Basic settings	*Mode*	Dans cette liste déroulante, sélectionnez : -**File** pour valider un fichier d'entrée, -**Flow** pour valider un flux d'entrée.
	Schema et *Edit Schema*	Un schéma est une description de lignes, i.e., il définit le nombre de champs qui sont traités et passés au composant suivant. Le schéma est soit local (built-in) soit distant dans le Repository. Le schéma de ce composant est en lecture seule. Il contient les informations standard de validation de fichier.
Mode File uniquement	*XSD file*	Chemin d'accès au fichier XSD de référence.
Mode File uniquement	*XML file*	Chemin d'accès au fichier XML à valider.
Mode File uniquement	*If XML is valid, display* *If XML is not valid detected, display*	Saisissez un message à afficher dans la console **Run** basé sur le résultat de la comparaison.
Mode File uniquement	*Print to console*	Cochez cette case pour afficher le message de validation.
Mode Flow uniquement	*Allocate*	Spécifiez la ou les colonne(s) à valider ainsi que le chemin d'accès au fichier XSD de référence.
Advanced settings	*tStatCatcher Statistics*	Cochez cette case pour collecter les données de log au niveau du composant.
Utilisation	Ce composant peut être utilisé en standalone mais il est généralement connecté à un composant de sortie pour rassembler les informations de validation.	
Limitation	n/a	

Scénario : Valider des flux de données par rapport à un fichier XSD

Ce scénario décrit un Job en Java qui valide une colonne XML dans un fichier d'entrée par rapport à un fichier XSD de référence et écrit les informations de log concernant les lignes invalides de la colonne dans un fichier délimité.

Pour un exemple d'utilisation du tXSDValidator validant un fichier XML, consultez le *Scénario : Valider un fichier XML, page 1451*.

- Déposez de la **Palette** dans l'espace de modélisation graphique les composants **tFileInputDelimited**, **tXSDValidator** ainsi que deux **tFileOutputDelimited**.

- Double-cliquez sur le composant **tFileInputDelimited** afin d'ouvrir sa vue **Component** et définir ses propriétés.

- Dans la liste **Property Type**, sélectionnez **Built-In**.
- Dans le champ **File name/Stream**, cliquez sur le bouton **[...]** afin de parcourir votre répertoire jusqu'au fichier d'entrée.
- Dans le champ **Header**, définissez le nombre de lignes à ignorer au début du fichier.
- Dans la liste **Schema**, sélectionnez **Built-In**.
- Cliquez sur **Edit Schema** et modifiez le schéma pour qu'il corresponde à celui du fichier d'entrée. Dans ce scénario, le fichier d'entrée contient deux colonnes : *ID* et *ShipmentInfo*. La colonne *ShipmentInfo* est une colonne XML nécessitant d'être validée.

- Dans l'espace de modélisation graphique, reliez le composant **tFileInputDelimited** au **tXSDValidator** à l'aide d'un lien **Row** > **Main**.
- Double-cliquez sur le composant **tXSDValidator** et configurez ses propriétés.

- Dans la liste **Mode**, sélectionnez **Flow Mode**.
- Dans la liste **Schema**, sélectionnez **Built-In**. Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent.
  Pour voir ou modifier le schéma, cliquez sur le bouton **[...]**.
- Cliquez sur le bouton **[+]** sous le tableau **Allocate** afin d'ajouter une ligne. Le nom de la première colonne du fichier d'entrée apparaît automatiquement dans le champ **Input Column**. Cliquez dans le champ et sélectionnez la colonne que vous souhaitez valider.
- Dans le champ **XSD File**, saisissez le chemin d'accès à votre fichier XSD de référence.
- Dans l'espace de modélisation, reliez le composant **tXSDValidator** au **tFileOutputDelimited** à l'aide d'un lien **Row** > **Main** afin d'écrire les informations concernant les lignes XML valides.
- Reliez le **tXSDValidator** au second **tFileOutputDelimited** à l'aide d'un lien **Row** > **Rejects** afin d'écrire les informations concernat les lignes XML invalides.

- Double-cliquez sur les composants **tFileOutputDelimited** l'un après l'autre et configurez leurs propriétés.
- Dans le champ **Property Type**, sélectionnez **Built-In**.
- Dans le champ **File Name**, parcourez votre répertoire jusqu'au fichier que vous souhaitez utiliser.
- Dans la liste **Schema**, sélectionnez **Built-In** puis cliquez sur le bouton **Sync columns** afin de récupérer le schéma du composant précédent.

- Sauvegardez votre Job et appuyez sur **F6** pour l'exécuter.

Les fichiers de sortie contiennent les informations concernant les lignes XML valides et invalides de la colonne spécifiée.

tXSLT

Propriétés du tXSLT

Famille de composant	XML	
Fonction	Le composant **tXSLT** tient compte d'une feuille de style XSL pour transformer un fichier source XML et l'envoyer vers la sortie définie.	
Objectif	Le **tXSLT** permet de transformer une structure de données en une autre structure.	
Basic settings	*XML file*	Chemin d'accès au fichier XML à transformer.
	XSL file	Chemin d'accès au fichier XSL de référence.
	Output file	Chemin d'accès du fichier de sortie. Si le fichier n'existe pas, il sera créé. Le fichier de sortie peut être n'importe quel fichier structuré ou non tels que html, xml, txt ou même pdf ou edifact en fonction de votre fichier xsl.
	Parameters	Cliquez sur le bouton **[+]** pour ajouter de nouvelles lignes à la liste **Parameters** et définir les paramètres de transformation du fichier XSLT. Pour chaque ligne, renseignez la clé dans la colonne *name* et sa valeur associée dans la colonne *value*.
Utilisation	Ce composant peut être utilisé en standalone.	
Limitation	n/a	

Scénario : Transformer un fichier XML en HTML à l'aide d'une feuille de style XSL

Ce scénario décrit un Job à deux composants appliquant une feuille de style XSL à un fichier XML et génère un fichier HTML en sortie. Il permet également de définir un paramètre de transformation pour la feuille de style XSL afin de changer la couleur de l'arrière plan de l'en-tête dans le document HTML créé en sortie.

- A partir de la **Palette**, cliquez et déposez un composant **tXSLT** et un **tMsBox** dans l'espace graphique.

- Double-cliquez sur le **tXSLT** pour ouvrir la vue **Basic settings** et paramétrer le composant.

- Dans le champ **XML file**, renseignez le chemin d'accès ou parcourez vos dossiers jusqu'au fichier XML à transformer. Dans cet exemple, il s'agit d'une liste de titres MP3 et leurs informations associées (nom de l'artiste, maison de disque...).

```
1   <?xml version="1.0" encoding="ISO-8859-1"?>
2   <catalog>
3    <cd>
4       <title>Empire Burlesque</title>
5       <artist>Bob Dylan</artist>
6       <country>USA</country>
7       <company>Columbia</company>
8       <price>10.90</price>
9       <year>1985</year>
10    </cd>
11    .
12    .
13
14  </catalog>
```

- Dans le champ **XSL file** de la vue **Basic settings**, renseignez le chemin d'accès ou parcourez vos dossiers jusqu'au fichier XSL approprié.
- Dans le champ **Output file**, renseignez le chemin d'accès ou parcourez vos dossiers jusqu'au fichier HTML.

Dans cet exemple, l'objectif est de convertir les données XML en fichier HTML comprenant un tableau constitué d'un en-tête suivi de la liste du nom des artistes à côté du nom des chansons.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:param name="bgcolor" />

<xsl:template match="/">
  <html>
  <body>
  <h2>My CD Collection</h2>
  <table border="1">
    <tr bgcolor="{$bgcolor}">
      <th>Title</th>
      <th>Artist</th>
    </tr>
    <xsl:for-each select="catalog/cd">
    <tr>
      <td><xsl:value-of select="title"/></td>
      <td><xsl:value-of select="artist"/></td>
    </tr>
    </xsl:for-each>
  </table>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

- Dans la zone **Parameters** de la vue **Basic settings**, cliquez sur le bouton **[+]** pour ajouter une ligne où définir le nom et la valeur du paramètre de transformation du fichier XSL. Dans cet exemple, le nom du paramètre de transformation utilisé est *bgcolor* et sa valeur est *green*.
- Double-cliquez sur le **tMsgBox** pour afficher la vue **Basic settings** et paramétrer le composant.

- Enregistrez votre Job puis appuyez sur **F6** pour l'exécuter. Une boîte de dialogue s'affiche et vient confirmer que le fichier HTML est bien créé en sortie et qu'il est stocké à l'endroit préalablement défini.

- Cliquez sur **Ok** pour fermer la boîte de dialogue.

Vous pouvez désormais ouvrir le fichier HTML créé pour vérifier la transformation des données XML et la couleur d'arrière-plan de l'en-tête du tableau.

My CD Collection

Title	Artist
Empire Burlesque	Bob Dylan
Hide your heart	Bonnie Tyler
Greatest Hits	Dolly Parton
Still got the blues	Gary Moore

A

Alias 839

B

Business

tBonitaDeploy xxxviii
tBonitaInstantiateProcess xl
tCentricCRMInput xlvi
tCentricCRMOutput xlvii
tHL7Input xlviii
tHL7Output liii
tMicrosoftCRMInput liv
tMicrosoftCRMOutput lxii
tMSAXInput lxiv
tMSAXOutput lxv
tOpenbravoERPInput lxxii
tOpenBravoERPOutput lxxiv
tSalesforceBulkExec lxxv
tSalesforceConnection lxxvii
tSalesforceGetDeleted lxxviii
tSalesforceGetServerTimestamp lxxxii
tSalesforceGetUpdated lxxxiv
tSalesforceInput lxxxvi
tSalesforceOutput xcii
tSalesforceOutputBulk xcvi
tSalesforceOutputBulkExec cii
tSAPCommit cvii
tSAPRollback cxxv
tSugarCRMInput cxxvi
tSugarCRMOutput cxxix
tVtigerCRMInput cxxx
tVtigerCRMOutput cxxxii

Business Intelligence

tDB2SCD 2
tIGreenplumSCD 7
tInformixSCD 9
tIngresSCD 11
tMondrianInput 13
tMSSqlSCD 17
tMysqlSCD 19
tOracleSCD 36
tPaloCheckElements 40
tPaloConnection 43
tPaloCube 44
tPaloCubeList 48
tPaloDatabase 52
tPaloDatabaseList 55
tPaloDimension 59
tPaloDimensionList 68
tPaloInputMulti 72
tPaloOutput 78
tPaloOutputMulti 80
tPaloRule 89, 93
tParAccelSCD 97
tSPSSInput 109
tSPSSOutput 113
tSPSSProperties 116
tSPSSStructure 117
tSybaseSCD 119

BusinessIntelligence

PostgresPlusSCD 99
PostgresqlSCD 104

C

Component, Composant ..... 1, 143, 177, 883, 1247, ... 1371
Contexte 1354

Custom Code

tJava 126, 127, 129
tJavaFlex 132, 139, 140

CustomCode

tGroovy 126
tGroovyFile 127
tJava 129
tJavaFlex 132
tJavaRow 139
tLibraryLoad 140

D

Data Quality

tAddCRCRow 144
tFuzzyMatch 148
tIntervalMatch 154
tParseAddress 159
tParseName 161
tReplaceList 163
tSchemaComplianceCheck 168
tUniqRow 173

Data quality

tAddCRCRow 144, 159

Database
PostgresPlusOutput 646
tAccessBulkExec 178
tAccessInput 181
tAccessOutput 183
tAccessOutputBulk 187
tAccessOutputBulkExec 189
tAccessRow 191
tAS400Close 193
tAS400Commit 194
tAS400Connection 195
tAS400Input 197
tAS400LastInsertId 200
tAS400Output 201
tAS400Rollback 206
tAS400Row 207
tDB2BulkExec 215
tDB2Close 218
tDB2Commit 219
tDB2Connection 220
tDB2Input 222
tDB2Output 225
tDB2Rollback 230
tDB2Row 231
tDB2SCD 234, 1035
tDB2SCDELT 4, 235
tDB2SP 236
tDBInput 238, 929
tDBOutput 243
tDBSQLRow 248
tEXAInput 252
tEXAOutput 255
tEXARow 258
tEXistConnection 260
tFirebirdClose 274
tFirebirdCommit 275
tFirebirdConnection 276
tFirebirdInput 278
tFirebirdOutput 280
tFirebirdRollback 283
tFirebirdRow 284
tFixedFlowInput 1193
tGreenplumBulkExec 287
tGreenplumClose 290
tGreenplumCommit 291
tGreenplumConnection 292
tGreenplumInput 294
tGreenplumOutput 297
tGreenplumOutputBulk 301
tGreenplumOutputBulkExec 303
tGreenplumRollback 306
tGreenplumRow 307
tGreenplumSCD 310
tHiveClose 311
tHiveConnection 312
tHiveRow 314
tHSQLDbInput 317
tHSQLDbOutput 321
tHSQLDbRow 326
tInformixBulkExec 329
tInformixClose 333
tInformixCommit 334
tInformixConnection 335
tInformixInput 337
tInformixOutput 340
tInformixOutputBulk 344
tInformixOutputBulkExec 346
tInformixRollback 350
tInformixRow 351
tInformixSCD 354
tInformixSP 355
tIngresClose 358
tIngresCommit 359
tIngresConnection 360
tIngresInput 361
tIngresOutput 364
tIngresRollback 367
tIngresRow 368
tIngresSCD 370
tInterbaseClose 371
tInterbaseCommit 372
tInterbaseConnection 373
tInterbaseInput 374
tInterbaseOutput 376
tInterbaseRollback 379
tInterbaseRow 380
tJavaDBInput 383
tJavaDBOutput 385
tJavaDBRow 388
tJDBCClose 391

tJDBCColumnList 390
tJDBCCommit 392
tJDBCConnection 393
tJDBCInput 395
tJDBCOutput 398
tJDBCRollback 402
tJDBCRow 403
tJDBCSP 406
tJDBCTableList 408
tLDAPAttributesInput 409
tLDAPInput 412
tLDAPOutput 417
tLDAPRenameEntry 422
tMaxDBInput 424
tMaxDBOutput 426
tMaxDBRow 429
tMSSqlBulkExec 431
tMSSqlClose 434
tMSSqlCommit 436
tMSSqlConnection 437
tMSSqlInput 439
tMSSqlLastInsertId 442
tMSSqlOutput 442, 443
tMSSqlOutputBulk 448
tMSSqlOutputBulkExec 450
tMSSqlRollback 453
tMSSqlRow 454
tMSSqlSCD 457
tMSSqlSP 458
tMSSqlTableList 461
tMysqlBulkExec 287, 462, 546, 611
tMysqlClose 465
tMysqlColumnList 466
tMysqlCommit .... 193, 218, 219, 274, 275, 290, . 291, 311, 333, 334, 358, 359, 371, . 372, 390, 391, 392, 408, 434, 435, . 436, 461, 465, 466, 470, 541, 549, . 550, 578, 579, 610, 614, 615, 639, . 640, 664, 665, 693, 694, 713, 714, . 744, 745, 775, 792, 793, 819, 1093
tMysqlConnection 195, 220, 260, 276, 292, . 312, 335, 360, 373, 393, 437, 471, 551, . 580, 616, 641, 666, 695, 715, 746, ... 776, 794
tMysqlInput 476, 796
tMysqlLastInsertId 484
tMysqlOutput 489, 799
tMysqlOutputBulk 513
tMysqlOutputBulkExec 518
tMysqlRollback .cxxv, 200, 206, 230, 283, 306, ..350, 367, 379, 453, 484, 522, 567, ..630, 655, 705, 735, 765, 785, 807, ...870
tMysqlRow 524, 568
tMysqlSCD 534
tMysqlSCDELT 32, 535
tMysqlSP 536
tMysqlTableList 541
tNetezzaBulkExec 546
tNetezzaClose 549
tNetezzaCommit 550
tNetezzaConnection 551
tNetezzaInput 553
tNetezzaNzLoad 556
tNetezzaOutput 563
tNetezzaRollback 567
tNetezzaRow 568
tOracleBulkExec 571
tOracleClose 578
tOracleCommit 579
tOracleConnection 580
tOracleInput 582
tOracleOutput 585
tOracleOutputBulk 591
tOracleOutputBulkExec 593
tOracleRollback 597
tOracleRow 598
tOracleSCD 35, 601
tOracleSCDELT 37, 602
tOracleSP 603
tOracleTableList 610
tParAccelBulkExec 611
tParAccelClose 614
tParAccelCommit 615
tParAccelConnection 616
tParAccelInput 618
tParAccelOutput 621
tParAccelOutputBulk 625
tParAccelOutputBulkExec 627

tParAccelRollback 630
tParAccelRow 631
tParAccelSCD 634
tPosgtresqlOutput 671
tPostgresPlusBulkExec 636
tPostgresPlusClose 639
tPostgresPlusCommit 640
tPostgresPlusConnection 641
tPostgresPlusInput 643
tPostgresPlusOutputBulk 651
tPostgresPlusOutputBulkExec 653
tPostgresPlusRollback 655
tPostgresPlusRow 656
tPostgresPlusSCD 659
tPostgresPlusSCDELT 101, 660
tPostgresqlBulkExec 661
tPostgresqlClose 664
tPostgresqlCommit 665
tPostgresqlConnection 666
tPostgresqlInput 668
tPostgresqlOutputBulk 676
tPostgresqlOutputBulkExec 678
tPostgresqlRollback 681
tPostgresqlRow 682
tPostgresqlSCD 685
tPostgresqlSCDELT 106, 686
tSASInput 687
tSASOutput 690
tSQLiteClose 693
tSQLiteCommit 694
tSQLiteConnection 695
tSQLiteInput 697
tSQLiteOutput 701
tSQLiteRollback 705
tSQLiteRow 706
tSybaseBulkExec 710
tSybaseClose 713
tSybaseCommit 714
tSybaseConnection 715
tSybaseInput 716
tSybaseIQBulkExec 719
tSybaseIQOutputBulkExec 722
tSybaseOutput 725
tSybaseOutputBulk 730
tSybaseOutputBulkExec 732
tSybaseRollback 735
tSybaseRow 736
tSybaseSCD 739
tSybaseSCDELT 121, 740
tSybaseSP 741
tTeradataClose 744
tTeradataCommit 745
tTeradataConnection 746
tTeradataFastExport 748
tTeradataFastLoad 750
tTeradataFastLoadUtility 752
tTeradataInput 754
tTeradataMultiLoad 757
tTeradataOutput 760
tTeradataRollback 765
tTeradataRow 766
tTeradataTPump 769
tVectorWiseCommit 775
tVectorWiseConnection 776
tVectorWiseInput 778
tVectorWiseOutput 781
tVectorWiseRollback 785
tVectorWiseRow 786
tVerticaBulkExec 789
tVerticaClose 792
tVerticaCommit 275, 775, 793, 819
tVerticaConnection 794
tVerticaInput 796
tVerticaOutput 799
tVerticaOutputBulk 803
tVerticaOutputBulkExec 805
tVerticaRollback 807
tVerticaRow 808
Database Utilities
tCreateTable 210
tParseRecordSet 635
Databases
tEXistDelete 262
tEXistGet 264
tEXistList 266
tEXistPut 268
tEXistXQuery 270
tEXistXUpdate 272
DataQuality
tIntervalMatch 154

E

ELT

tELTAggregate 812
tELTCommit 819
tELTFilterColumns 821
tELTFilterRows 823
tELTJDBCInput 825
tELTJDBCMap 827
tELTJDBCOutput 829
tELTMSSqlInput 831
tELTMSSqlMap 833
tELTMSSqlOutput 835
tELTMysqlInput .. 825, 831, 837, 864, 872
tELTMysqlMap ... 827, 833, 838, 866, 874
tELTMysqlOutput 829, 835, 847, 868, 876
tELTOracleInput 849
tELTOracleMap 851
tELTOracleOutput 856
tELTPostgresqlInput 864
tELTPostgresqlMap 866
tELTPostgresqlOutput 868
tELTRollback 870
tELTSybaseInput 872
tELTSybaseMap 874
tELTSybaseOutput 876
tELTTeradataInput 878
tELTTeradataMap 879
tELTTeradataOutput 881

F

File

tApacheLogInput 885
tCreateTemporaryFile 888
tExtractPositionalFields 1281
tFileArchive 893
tFileCompare 896
tFileCopy 900
tFileDelete 903, 906
tFileExist 906
tFileInputARFF 911
tFileInputDelimited ..... 885, 911, 915, 932, 938, . 943, 1014, 1016, 1379, 1384, 1389, ... 1399, 1423, 1428
tFileInputExcel 929
tFileInputFullRow 932
tFileInputJSON 935, 990
tFileInputLDIF 938
tFileInputMail 940
tFileInputPositional .....950, 956, 961, 998, 1000, ...1009
tFileInputRegex 965
tFileInputXML ..935, 952, 969, 990, 1001, 1454, ...1462, 1472
tFileList ...147, 888, 970, 1042, 1210, 1229
tFileOutputARFF 976
tFileOutputDelimited 978
tFileOutputExcel 988, 1011, 1047
tFileOutputLDIF 994
tFileOutputMSDelimited 998
tFileOutputMSPositional 1000
tFileOutputPositional 1007
tFileOutputXML 1010, 1470
tFileProperties 1011
tFileRowCount 1014
tFileTouch 1016
tFileUnarchive 893, 1017, 1019
tFTPFileProperties 1047
tGPGDecrypt 1019
tPivotToColumnsDelimited 1023

I

Internet

tFileFetch 1028
tFTPConnection 1036
tFTPDelete 1038, 1040
tFTPFileExist 1040
tFTPFileList 1042
tFTPGet 1049
tFTPPut .....262, 264, 266, 268, 270, 272, 1036, ...1052, 1056, 1058
tFTPRename 1056
tFTPTruncate 1058
tJMSInput 1060
tJMSOutput 1062
tMicrosoftMQInput 1063
tMicrosoftMQOutput 1067
tMomInput 1068
tMomMessageIdList 1072
tMomOutput 1073
tPOP 1074

tRSSInput 1078
tRSSOutput 1081
tSCPClose 1093
tSCPConnection 1094
tSCPDelete 1095
tSCPFileExists 1096
tSCPFileList 1094, 1097
tSCPGet 1098
tSCPPut 1100
tSCPRename 1101
tSCPTruncate 1102
tSendMail 1103
tSetKeyStore 1108
tSOAP 1121
tSocketInput 1114
tSocketOutput 1119
tWebServiceInput 1125
tXMLRPCInput 1134

J

Jointure
Explicite 839
Jointure explicite 839
Jointures
Left Outer Join 845

L

Log&Error
tDie 1152
tLogCatcher 1159
tLogRow 1163
Logs&Errors
tAssert 1138
tAssertCatcher 1144
tChronometerStart 1146
tChronometerStop 1147
tFlowMeter 1153
tFlowMeterCatcher 1154
tStatCatcher 1164
tWarn 1167

M

Misc
tAddLocationFromIP 1170
tAlfrescoOutput xxvi
tBufferOutput 1174, 1177
tContextDump 1188
tContextLoad 1189
tLoop 1226, 1227
tMemorizeRows 1195
tMsgBox 1202
tRowGenerator 1204
MultiSchema
tFileInputMSDelimited 943
tFileInputMSPositional 950
tFileInputMSXML 952

O

Orchestration
tFileList 1210
tFlowToIterate 1211
tForeach 1216
tInfiniteLoop 1219
tIterateToFlow 1220
tLoop 1223
tPostjob 1226
tPrejob 1227
tReplicate 1228
tRunJob 1229
tSleep 1230
tUnite 1231
tWaitForFile 1235
tWaitForSocket 1240
tWaitForSqlData 1242

P

Processing
tAggregateRow 812, 1248
tAggregateSortedRow 1254
tConvertType 1256
tDenormalize 1261
tDenormalizeSortedRow 1266
tEmptyToNull 1270
tExternalSortedRow 1274
tExtractDelimitedFields 1276
tExtractRegexFields 1283
tExtractXMLField 1287
tFilterColumns 1288
tFilterRow 1289
tMap 1294

tNormalize 1328
tPerl 1331
tPivotToRows 1334
tReplace 1338
tSampleRow 1343
tSortRow 1346
Propriétés du tDB2Close ... 218, 274, 290, 311

S

StoreSQLQuery 241
System
tRunJob 1352
tSetEnv 1359
tSSH 1363
tSystem 1367

T

Table
Alias 839
tAccessInput
Advanced settings 182
tAccessOutput
Advanced settings 184
Talend MDM
tMDMBulkLoad 1372
tMDMDelete 1379
tMDMInput 1384
tMDMOutput 1389
tMDMReceive 1397
tMDMRouteRecord 1399
tMDMSP 1410
tMDMViewSearch 1417
tStewardshipTaskInput 1423
tStewardshipTaskOutput 1428
tAlfrescoOutput
Advanced settings xxvii
tAS400Connection
Advanced settings 195
tAS400Input
Advanced settings 198, 425
tAS400Output
Advanced settings 204
tDB2Input
Advanced settings .. 224, 1190, 1193, 1204
tDB2Output
Advanced settings 6, 103, 108, 217, 228
tDBInput
Advanced settings 239
tDBOutput
Advanced settings 245
tFileInputEBCDIC 923
tFileInputExcel
Advanced settings 930
tFileOutputEBCDIC 985
tFirebirdInput
Advanced settings 253, 259, 279
tFirebirdOutput
Advanced settings 256, 281
tFTPFileList 1042
tHSQLDBOutput
Advanced settings 427
tHSQLDbOutput
Advanced settings 427
tInformixOutput
Advanced settings 331, 338, 342
tIngresInput
Advanced settings 319, 324, 327, 362
tIngresOutput
Advanced settings 365
tInterbaseOutput
Advanced settings 378
tJavaDBInput
Advanced settings 384
tJavaDBOutput
Advanced settings 386, 400
tJDBCInput
Advanced settings 397
tJDBCOutput
Advanced settings 400
tMSSqlInput
Advanced settings 440
tMSSqlOutput
Advanced settings 446
tMysqlInput
Advanced settings 478
tMysqlOutput
Advanced settings 331, 338, 342
tNetezzaNzLoad
Advanced settings 557
tOracleInput

Advanced settings 584
tOracleOutput
Advanced settings 588
tPostegresqlInput
Advanced settings 295, 620, 645, 670
tPostegresqlOutput
Advanced settings 299, 612, 623, 649, 674
tPostegrsqlOutput
Advanced settings 299, 612, 623, 649, 674
tSASInput
Advanced settings 688
tSASOutput
Advanced settings 692
tSQLiteInput
Advanced settings 698
tSQLiteOutput
Advanced settings 703
tSybaseInput
Advanced settings 718
tSybaseOutput
Advanced settings 728
tTeradataInput
Advanced settings .748, 750, 753, 758, 770
tTeradataOutput
Advanced settings 755, 763, 767, 780, 783, ...787

V

Variable 1354
StoreSQLQuery 241

X

XML
tAdvancedFileOutputXML 1440
tDTDValidator 1451
tExtractXMLField 1454
tFileOutputMSXML 1001
tWriteXMLField 1472
tXSDValidator 1478
tXSLT 1482