REST Exercise

This exercise objective on implementation of REST API including with JDBC connect database and manipulate data with CRUD.

Table schema

**CREATE** **TABLE** ttransaction (

transid **VARCHAR**(50) **NOT** **NULL** **COMMENT** 'UUID',

userid **VARCHAR**(50) **NOT** **NULL** **COMMENT** 'tuser.userid',

transdate **DATE** **NOT** **NULL**,

journalid **VARCHAR**(50) **NOT** **NULL** **COMMENT** 'tjournal.journalid',

transno **INT**(11) **NOT** **NULL** **COMMENT** 'sequence number',

amount **DECIMAL**(20,4) **NOT** **NULL**,

remark **VARCHAR**(200) **NULL** **DEFAULT** **NULL**,

transtime **DATETIME** **NULL** **DEFAULT** **NULL**,

editdate **DATE** **NULL** **DEFAULT** **NULL**,

edittime **TIME** **NULL** **DEFAULT** **NULL**,

**PRIMARY** **KEY** (transid),

**UNIQUE** **INDEX** userid\_trxdate\_journalid (userid, transdate, journalid)

);

**CREATE** **TABLE** tjournal (

journalid **VARCHAR**(50) **NOT** **NULL** **COMMENT** 'UUID',

journalname **VARCHAR**(100) **NOT** **NULL**,

journaltype **VARCHAR**(1) **NOT** **NULL** **DEFAULT** '0' **COMMENT** '0=Payment,1=Receive',

amount **DECIMAL**(20,4) **NOT** **NULL** **DEFAULT** '0.0000',

editdate **DATE** **NULL** **DEFAULT** **NULL**,

edittime **TIME** **NULL** **DEFAULT** **NULL**,

**PRIMARY** **KEY** (journalid)

);

Specification

1. Create REST API
   1. input format can be query string or json string format
   2. output is json format like below

{

"head": {

"model": "transaction",

"method": "create",

"errorflag": "Y",

"errorcode": "400",

"errordesc": "Not accept"

},

"body": {

}

}

* 1. head

model = class name or model name

method = method name

errorflag = Y is error and N if not error

errorcode = user defined error code

errordesc = user defined error description

* 1. body is user defined data attributes format

1. CRUD
   1. defined resource name = transaction
   2. create : try to insert new record with
      1. path = create
      2. http method = POST
      3. transid = UUID if not specified
      4. transdate = current date if not specified
      5. userid = ‘tester’
      6. journalid = Transaction from parameter
      7. transno = 1
      8. amount = Amount from parameter
      9. remark = Remark from parameter
      10. transtime = current timestamp
      11. editdate = current date
      12. edittime = current time

ex. request: json string

{ "journalid":"1001", "amount": 100, "remark":"Noodle" }

Example response

{

"head": {

"model": "transaction",

"method": "create",

"errorflag": "N",

"errorcode": "0",

"errordesc": ""

},

"body": {

"transid": "498e18b8-ac0f-11ed-809e-98fa9bd6bd8e"

}

}

* 1. retrieve : try to retreive record with
     1. path = retrieve
     2. http method = GET
     3. using path param = transid
     4. query statement

select ttransaction.\*,tjournal.journalname,tjournal.journaltype

from ttransaction,tjournal

where ttransaction.transid = ?

and ttransaction.journalid = tjournal.journalid

ex. request: with transid = 498e18b8-ac0f-11ed-809e-98fa9bd6bd8e

http://localhost:8080/transaction/retrieve/498e18b8-ac0f-11ed-809e-98fa9bd6bd8e

Example response

{

"head": {

"model": "transaction",

"method": "retrieve",

"errorflag": "N",

"errorcode": "0",

"errordesc": ""

},

"body": {

"transid": "498e18b8-ac0f-11ed-809e-98fa9bd6bd8e",

"transdate": "2023-02-01",

"journalid": "1001",

"transno": 1,

"amount": 120.00,

"remark": "Noodle + Coffee",

"journalname": "Breakfast",

"journaltype": "0"

}

}

* 1. update : try to update record with
     1. path = update
     2. http method = PUT
     3. transid = from parameter
     4. journalid = from parameter
     5. amount = from input
     6. remark = from input
     7. editdate = current date
     8. edittime = current time
     9. query statement

update ttransaction set journalid=?, amount=?, remark=? where transid=?

ex. request: json string

{ "transid": " 498e18b8-ac0f-11ed-809e-98fa9bd6bd8e", "journalid": "1001", "amount": 80, "remark": "Noodle" }

Example response

{

"head": {

"model": "transaction",

"method": "update",

"errorflag": "N",

"errorcode": "0",

"errordesc": ""

},

"body": {

}

}

* 1. delete : try to delete record with
     1. path = delete
     2. http method = DELETE
     3. using query param = transid
     4. transid = from parameter
     5. query statement

delete from ttransaction where transid=?

ex. request: transid = 498e18b8-ac0f-11ed-809e-98fa9bd6bd8e

http://localhost:8080/transaction/delete?transid=498e18b8-ac0f-11ed-809e-98fa9bd6bd8e

Example response

{

"head": {

"model": "transaction",

"method": "delete",

"errorflag": "N",

"errorcode": "0",

"errordesc": ""

},

"body": {

}

}

* 1. find: try to find out all records specified by transaction date with
     1. path = find
     2. http method = POST
     3. query statement

select ttransaction.\*,tjournal.journalname,tjournal.journaltype

from ttransaction,tjournal

where ttransaction.transdate = ?

and ttransaction.journalid = tjournal.journalid

order by transno

* + 1. transdate = from parameter with format dd/MM/yyyy or yyyy-MM-dd

ex. request: json string

{ "transdate": "2023-02-01" }

Example response

{

"head": {

"model": "transaction",

"method": "find",

"errorflag": "N",

"errorcode": "0",

"errordesc": ""

},

"body": {

"transdate": "2023-02-01",

"records": [ {

"transid": "498e18b8-ac0f-11ed-809e-98fa9bd6bd8e",

"transno": 1,

"amount": 120.00,

"remark": "Noodle + Coffee",

"journalid": "1001",

"journalname": "Breakfast",

"journaltype": "0"

}, {

"transid": "5055f401-ac0f-11ed-809e-98fa9bd6bd8e",

"transno": 2,

"amount": 100.00,

"remark": "Fry Rice" ,

"journalid": "1002",

"journalname": "Lunch",

"journaltype": "0"

}, {

"transid": "57781363-ac0f-11ed-809e-98fa9bd6bd8e",

"transno": 3,

"amount": 250.00,

"remark": "Steak",

"journalid": "1003",

"journalname": "Dinner",

"journaltype": "0"

}, {

"transid": "44d814de-ac10-11ed-809e-98fa9bd6bd8e",

"transno": 3,

"amount": 1100.00,

"remark": null,

"journalid": "2002",

"journalname": "Commission",

"journaltype": "1"

}]

}

}