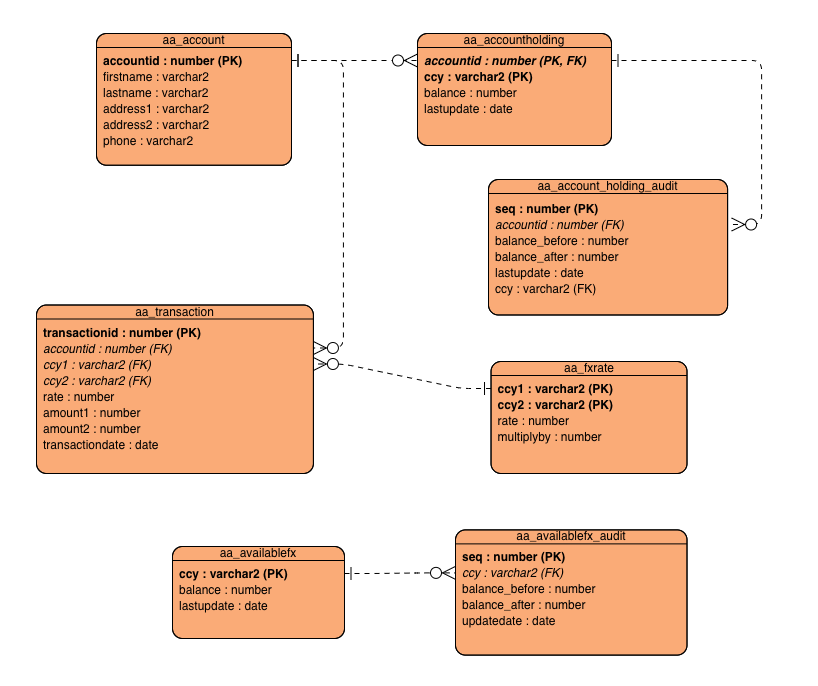
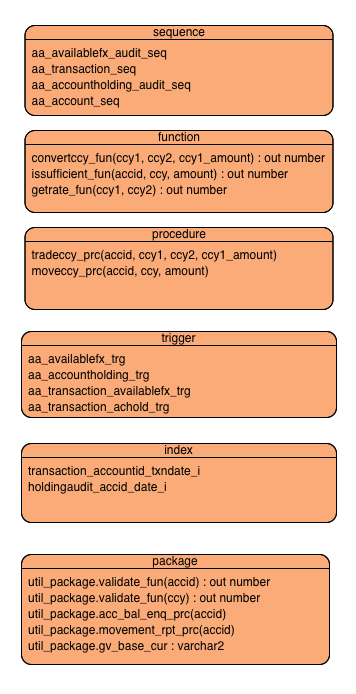
Title: ON9 FX System

Business: Foreign currencies exchange services

ERD:



Func:



Customer Fx rate ***Enquiry*** (SQL)

Customer account ***Enquiry*** (SQL + function rateConvert)

Customer ***PlaceOrder*** (procedure (insert to transaction table) + ***Trigger*** (update available balance)

***Withdraw*** currency (procedure with overloading)

Night batch ***Clearing*** (procedure update transaction status)

***AuditTrail*** (trigger) → generate a report table (transaction history)

Table:

CustomerAccount

CustomerTransaction

CurrencyRate

AvailableCurrency

AuditTrail

**Thomas:**

convertccy\_fun(ccy1, ccy2, amount) : out number

CREATE OR REPLACE

FUNCTION convertccy\_fun

(ccy1\_name IN VARCHAR2,

ccy2\_name IN VARCHAR2,

ccy1\_amount IN NUMBER)

RETURN NUMBER

IS

ccy2\_amount NUMBER(18, 8);

ccy\_rate NUMBER(18, 8);

ccy\_multiplyby NUMBER(18, 8);

BEGIN

IF ccy1\_name = ccy2\_name THEN

RETURN ccy1\_amount;

END IF;

SELECT rate, multiplyby INTO ccy\_rate, ccy\_multiplyby

FROM aa\_fxrate

WHERE ccy1\_name = ccy1 AND ccy2\_name = ccy2;

ccy2\_amount := ccy1\_amount \* ccy\_rate \* ccy\_multiplyby;

RETURN ccy2\_amount;

END;

tradeccy\_prc(ccy1, ccy2, amount, direction)

CREATE OR REPLACE

PROCEDURE tradeccy\_prc

(acc\_num IN NUMBER,

ccy1\_name IN VARCHAR2,

ccy2\_name IN VARCHAR2,

ccy1\_amount IN NUMBER)

AS

acc\_balance NUMBER(18, 8);

ccy\_rate NUMBER(18, 8);

ccy\_multiplyby NUMBER(18, 8);

ccy2\_amount NUMBER(18, 8);

BEGIN

IF issufficient\_fun(acc\_num, ccy1\_name, ccy1\_amount) = 1 Then

INSERT INTO AA\_TRANSACTION VALUES

(aa\_transaction\_seq.NEXTVAL, acc\_num, ccy1\_name, ccy2\_name, getrate\_fun(ccy1\_name, ccy2\_name),

ccy1\_amount, convertccy\_fun(ccy1\_name, ccy2\_name, ccy1\_amount), SYSDATE);

DBMS\_OUTPUT.PUT\_LINE('TRACNSACTION SUCCEED');

END IF;

END;

Transaction\_accountid\_txndate\_i (KW: done in schema)

**Noah:**

issufficient\_fun(accid, ccy, amount) : out bit or boolean

CREATE OR REPLACE

FUNCTION issufficient\_fun

(accid\_num IN NUMBER, ccy\_id IN VARCHAR2, amount\_num IN NUMBER)

RETURN NUMBER

IS

acc\_num aa\_accountholding.accountid%type;

ccy\_name aa\_accountholding.ccy%type;

holding\_bal aa\_accountholding.amount%type; --NUMBER(18, 8);

ex\_check\_available EXCEPTION;

BEGIN

SELECT accountid, ccy, amount INTO acc\_num, ccyID, holding\_bal

FROM aa\_acountholding

WHERE acc\_num = accid\_num AND ccy\_id = ccy\_name;

IF acc\_num IS NOT NULL THEN

IF amount\_num <= holding\_bal THEN

RETURN 1;

ELSE

RETURN 0;

END IF;

END IF;

EXCEPTION

WHEN ex\_check\_available THEN

DBMS\_OUTPUT.PUT\_LINE('The account provided is not existed!');

END;

moveccy\_prc(ccy, amount (+/-))

CREATE OR REPLACE PROCEDURE moveccy\_prc (

p\_ccy IN aa\_availablefx.ccy%TYPE,

p\_amount IN NUMBER

) AS

v\_balance\_before NUMBER(18, 8);

v\_balance\_after NUMBER(18, 8);

BEGIN

-- Retrieve the current available balance for the given currency

SELECT balance

INTO v\_balance\_before

FROM aa\_availablefx

WHERE ccy = p\_ccy;

-- Update the available balance for the given currency

UPDATE aa\_availablefx

SET balance = balance + p\_amount,

lastupdate = SYSDATE

WHERE ccy = p\_ccy;

-- Retrieve the updated available balance for the given currency

SELECT balance

INTO v\_balance\_after

FROM aa\_availablefx

WHERE ccy = p\_ccy;

-- Insert an audit record for the update

INSERT INTO aa\_availablefx\_audit (

seq,

ccy,

balance\_before,

balance\_after,

updatedate

) VALUES (

aa\_account\_sequence.NEXTVAL,

p\_ccy,

v\_balance\_before,

v\_balance\_after,

SYSDATE

);

-- Call issufficient\_fun to check if the balance is sufficient

IF NOT issufficient\_fun(NULL, p\_ccy, ABS(p\_amount)) THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Not enough funds available.');

END IF;

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20002, 'Invalid currency.');

WHEN OTHERS THEN

ROLLBACK;

RAISE;

END;

aa\_accountholding\_trg (KW: done in schema)

holdingaudit\_accid\_date\_i (KW: done in schema)

aa\_availablefx\_trg (KW: done in schema)

Kent:

drop package util\_package;

create or replace package util\_package as

procedure acc\_bal\_enq\_prc(accid in number);

end util\_package;

/

create or replace package body util\_package as

procedure acc\_bal\_enq\_prc(accid in number) is

cursor cur\_acc\_bal is

select

accountid, ccy, balance, convertccy\_fun(ccy, 'CAD', balance) as cad\_bal

from aa\_accountholding

where accountid = accid;

ccy aa\_accountholding.ccy%type;

balance aa\_accountholding.balance%type;

cad\_bal aa\_accountholding.balance%type;

begin

dbms\_output.put\_line('Account ID: ' || accid);

for bal\_rec in cur\_acc\_bal loop

dbms\_output.put\_line(bal\_rec.ccy ||

': ' || bal\_rec.balance ||

' (CAD: ' || bal\_rec.cad\_bal || ')');

end loop;

exception

when no\_data\_found then

dbms\_output.put\_line('No currency found!');

when others then

dbms\_output.put\_line('Error: ' || SQLERRM);

end acc\_bal\_enq\_prc;

end util\_package;

CREATE OR REPLACE

FUNCTION getrate\_fun

(ccy1\_name IN VARCHAR2,

ccy2\_name IN VARCHAR2)

RETURN NUMBER

IS

ccy\_rate NUMBER(18, 8);

BEGIN

select rate into ccy\_rate

from aa\_fxrate

where ccy1 = ccy1\_name and ccy2 = ccy2\_name;

RETURN ccy\_rate;

END;