作业五

**(1)定义视图v\_customeInfo，要求可以替换同名视图，列出所有客户相关信息，输出属性包括客户 编号(CUST\_ID)、完整联系地址(PROVICIAL\_REGION、CITY与ADDRESS三个字段拼接)、邮政编 码(POSTAL\_CODE)、客户名称(类型为I的客户输出在Individual表中对应的姓名，使用LAST\_NA ME与FIRST\_NAME拼接;类型为B的客户输出在business表中对应的NAME**

CREATE OR **REPLACE** VIEW v\_customeInfo AS

(

SELECT c.CUST\_ID, **CONCAT**(c.PROVICIAL\_REGION,c.CITY,c.ADDRESS) AS ADDR, c.POSTAL\_CODE, **CONCAT**(i.LAST\_NAME,i.FIRST\_NAME)

FROM customer AS c, individual AS i

WHERE c.CUST\_TYPE\_CD='I' AND c.CUST\_ID=i.CUST\_ID

)

UNION

(

SELECT c.CUST\_ID, **CONCAT**(c.PROVICIAL\_REGION,c.CITY,c.ADDRESS) AS ADDR, c.POSTAL\_CODE, b.NAME

FROM customer AS c, business AS b

WHERE c.CUST\_TYPE\_CD='B' AND c.CUST\_ID=b.CUST\_ID

)

**(2)定义视图v\_transInfo，从acc\_transaction中查出不在其开户行进行的交易记录信息(即 EXECUTION\_BRANCH\_ID不为该账户对应的OPEN\_BRANCH\_ID)，输出属性包括交易金额 (AMOUNT)、交易时间(TXN\_DATE)、交易类型(CD显示为现金存入，CW显示为现金取出，TF 显示为汇款转出，TT显示为汇款转入，IC显示为利息计入，LI显示为贷款拨入)、执行交易的分支机构 名称(EXECUTION\_BRANCH\_NAME)、开户分支银行名称(OPEN\_BRANCH\_NAME)**

CREATE VIEW v\_transInfo AS

(

SELECT t.AMOUNT, t.TXN\_DATE, '现金存入' AS TYPE , b1.NAME AS EXECUTION\_BRANCH\_NAME, b2.NAME AS OPEN\_BRANCH\_NAME

FROM acc\_transaction AS t, account AS a, branch AS b1, branch AS b2

WHERE t.ACCOUNT\_ID=a.ACCOUNT\_ID AND t.EXECUTION\_BRANCH\_ID <> a.OPEN\_BRANCH\_ID AND t.TXN\_TYPE\_CD='CD' AND t.EXECUTION\_BRANCH\_ID=b1.BRANCH\_ID AND a.OPEN\_BRANCH\_ID=b2.BRANCH\_ID

)

UNION

(

SELECT t.AMOUNT, t.TXN\_DATE, '现金取出' AS TYPE , b1.NAME AS EXECUTION\_BRANCH\_NAME, b2.NAME AS OPEN\_BRANCH\_NAME

FROM acc\_transaction AS t, account AS a, branch AS b1, branch AS b2

WHERE t.ACCOUNT\_ID=a.ACCOUNT\_ID AND t.EXECUTION\_BRANCH\_ID <> a.OPEN\_BRANCH\_ID AND t.TXN\_TYPE\_CD='CW' AND t.EXECUTION\_BRANCH\_ID=b1.BRANCH\_ID AND a.OPEN\_BRANCH\_ID=b2.BRANCH\_ID

)

UNION

(

SELECT t.AMOUNT, t.TXN\_DATE, '汇款转出' AS TYPE , b1.NAME AS EXECUTION\_BRANCH\_NAME, b2.NAME AS OPEN\_BRANCH\_NAME

FROM acc\_transaction AS t, account AS a, branch AS b1, branch AS b2

WHERE t.ACCOUNT\_ID=a.ACCOUNT\_ID AND t.EXECUTION\_BRANCH\_ID <> a.OPEN\_BRANCH\_ID AND t.TXN\_TYPE\_CD='TF' AND t.EXECUTION\_BRANCH\_ID=b1.BRANCH\_ID AND a.OPEN\_BRANCH\_ID=b2.BRANCH\_ID

)

UNION

(

SELECT t.AMOUNT, t.TXN\_DATE, '汇款转入' AS TYPE , b1.NAME AS EXECUTION\_BRANCH\_NAME, b2.NAME AS OPEN\_BRANCH\_NAME

FROM acc\_transaction AS t, account AS a, branch AS b1, branch AS b2

WHERE t.ACCOUNT\_ID=a.ACCOUNT\_ID AND t.EXECUTION\_BRANCH\_ID <> a.OPEN\_BRANCH\_ID AND t.TXN\_TYPE\_CD='TT' AND t.EXECUTION\_BRANCH\_ID=b1.BRANCH\_ID AND a.OPEN\_BRANCH\_ID=b2.BRANCH\_ID

)

UNION

(

SELECT t.AMOUNT, t.TXN\_DATE, '利息计入' AS TYPE , b1.NAME AS EXECUTION\_BRANCH\_NAME, b2.NAME AS OPEN\_BRANCH\_NAME

FROM acc\_transaction AS t, account AS a, branch AS b1, branch AS b2

WHERE t.ACCOUNT\_ID=a.ACCOUNT\_ID AND t.EXECUTION\_BRANCH\_ID <> a.OPEN\_BRANCH\_ID AND t.TXN\_TYPE\_CD='IC' AND t.EXECUTION\_BRANCH\_ID=b1.BRANCH\_ID AND a.OPEN\_BRANCH\_ID=b2.BRANCH\_ID

)

UNION

(

SELECT t.AMOUNT, t.TXN\_DATE, '贷款拨入' AS TYPE , b1.NAME AS EXECUTION\_BRANCH\_NAME, b2.NAME AS OPEN\_BRANCH\_NAME

FROM acc\_transaction AS t, account AS a, branch AS b1, branch AS b2

WHERE t.ACCOUNT\_ID=a.ACCOUNT\_ID AND t.EXECUTION\_BRANCH\_ID <> a.OPEN\_BRANCH\_ID AND t.TXN\_TYPE\_CD='LI' AND t.EXECUTION\_BRANCH\_ID=b1.BRANCH\_ID AND a.OPEN\_BRANCH\_ID=b2.BRANCH\_ID

)

**(3)为了备份数据或复制生产数据进行测试，经常会需要把数据从现有的表复制到新的(临时)表中， 可以用CREATE TABLE ... AS ...子句进行表的复制。创建存储过程(copyTransactionFrom)，该存储 过程接收一个参数:起始日期(fDATE)，功能是将acc\_transaction表中交易日期(TXN\_DATE)晚 于等于起始日期的记录全部信息复制到新表(acc\_transactionNew)中(该表如存在应先删除)。提 示:在存储过程中执行DDL语句需要特殊处理。**

CREATE DEFINER=`root`@`localhost` PROCEDURE `copyTransactionFrom`(IN fDATE datetime)

BEGIN

DROP TABLE IF EXISTS `acc\_transactionNew`;

CREATE TABLE **acc\_transactionNew** AS

SELECT \*

FROM acc\_transaction

WHERE acc\_transaction.TXN\_DATE>fDATE;

END

**(4)编写函数func\_vipid，接收输入参数after\_date，返回开户日期(OPEN\_DATE)晚于 after\_date的、且可用余额最多的账户对应的客户编号(CUST\_ID)。**

CREATE DEFINER=`root`@`localhost` FUNCTION `func\_vipid`(after\_date datetime) RETURNS int

BEGIN

DECLARE customer\_id int;

DECLARE max\_balance DECIMAL(12,4);

SELECT **MAX**(AVAIL\_BALANCE) INTO max\_balance

FROM ACCOUNT

WHERE OPEN\_DATE>after\_date;

SELECT CUST\_ID INTO customer\_id

FROM ACCOUNT

WHERE AVAIL\_BALANCE=max\_balance;

RETURN customer\_id;

END

**(5)编写函数func\_rand\_email，随机生成合法的邮箱地址，@之前为⻓度不超过50的字符串，字符 串中符号可包括a-z、A-Z、0-9;@之后为163.com、hotmail.com或ecnu.edu.cn。**

CREATE DEFINER=`root`@`localhost` FUNCTION `func\_rand\_email`() RETURNS varchar(100) CHARSET utf8mb4 COLLATE utf8mb4\_general\_ci

BEGIN

DECLARE content CHAR(62) DEFAULT 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789';

DECLARE email VARCHAR(100) DEFAULT '';

DECLARE suffix1 CHAR(8) DEFAULT '@163.com';

DECLARE suffix2 CHAR(12) DEFAULT '@hotmail.com';

DECLARE suffix3 CHAR(12) DEFAULT '@ecnu.edu.cn';

DECLARE prefix VARCHAR(50) DEFAULT '';

DECLARE prefix\_len INT;

DECLARE suffix\_id INT;

DECLARE i INT;

SET prefix\_len=**FLOOR**(1+**RAND**()\*50);

SET i=1;

WHILE i<=prefix\_len DO

SET prefix=**CONCAT**(prefix,**SUBSTRING**(content,**FLOOR**(1+**RAND**()\*62),1));

SET i=i+1;

END WHILE;

SET suffix\_id=**FLOOR**(1+**RAND**()\*3);

CASE

WHEN suffix\_id=1 THEN SET email=**CONCAT**(email,prefix,suffix1);

WHEN suffix\_id=2 THEN SET email=**CONCAT**(email,prefix,suffix2);

WHEN suffix\_id=3 THEN SET email=**CONCAT**(email,prefix,suffix3);

END CASE;

RETURN email;

END

**(6)编写函数func\_rand\_telnum，可随机生成⻓度为11为的手机号，开头为130 131 132 133 134 135 136 137 138 139 186 187 189 151 157。**

CREATE DEFINER=`root`@`localhost` FUNCTION `func\_rand\_telnum`() RETURNS varchar(11) CHARSET utf8mb4 COLLATE utf8mb4\_general\_ci

BEGIN

DECLARE head VARCHAR(150) DEFAULT '130,131,132,133,134,135,136,137,138,139,186,187,189,151,157';

DECLARE content CHAR(10) DEFAULT '0123456789';

DECLARE phone VARCHAR(11) DEFAULT **substring**(head, 1+**FLOOR**(**RAND**()\*15)\*4, 3);

DECLARE i int DEFAULT 1;

DECLARE **len** int DEFAULT LENGTH(content);

WHILE i<9 DO

SET i=i+1;

SET phone = **CONCAT**(phone, **substring**(content, **floor**(1 + **RAND**() \* **len**), 1));

END WHILE;

RETURN phone;

END