BASIC TASKS

Consider the database you created during last session (a small database simulating a friendly bank). Write

SQL queries to answer the following questions:

1. In total, how many transactions have been carried out at the bank?

**Select count(txinid)**

**from acc\_transaction;**

1. How many accounts of type ‘CHK’ are there at this bank?

**SELECT COUNT(\*) AS chk\_account\_count**

**FROM ACCOUNT**

**WHERE productCD = 'CHK';**

1. Produce a list of job titles and how many employees hold this position.

**SELECT title, COUNT(\*) AS employee\_count**

**FROM EMPLOYEE**

**GROUP BY title;**

1. Produce a list of Customers and the number of accounts they have.

**SELECT c.custId, c.firstName, c.lastName, COUNT(a.accountId) AS account\_count**

**FROM CUSTOMER c**

**LEFT JOIN ACCOUNT a ON c.custId = a.custId**

**GROUP BY c.custId, c.firstName, c.lastName;**

1. What is the total available balance for the customer; James Hadley (cust\_id = 1)?

**SELECT c.custId, c.firstName, c.lastName, SUM(a.availBalance) AS total\_available\_balance**

**FROM CUSTOMER c**

**LEFT JOIN ACCOUNT a ON c.custId = a.custId**

**GROUP BY c.custId, c.firstName, c.lastName;**

1. Produce a list of all customers and their total available balance.

**SELECT c.custId, c.firstName, c.lastName, SUM(a.availBalance) AS total\_available\_balance**

**FROM CUSTOMER c**

**LEFT JOIN ACCOUNT a ON c.custId = a.custId**

**GROUP BY c.custId, c.firstName, c.lastName;**

1. Write a query to list all account product types and the average available balance for each type

**SELECT p.name AS product\_type, AVG(a.availBalance) AS average\_available\_balance**

**FROM PRODUCT p**

**JOIN ACCOUNT a ON p.productTypeCd = a.productTypeCd**

**GROUP BY p.name;**

MEDIUM TASKS

8.Find the total available balance in customers’ accounts where the opening branch was the Woburn Branch.

**Select custid,sum(availbalance) from account where openbranch=2;**

9.Produce a list of account product types and the highest available balance for each.

**SELECT p.name AS product\_type, MAX(a.availBalance) AS highest\_available\_balance**

**FROM PRODUCT p**

**JOIN ACCOUNT a ON p.productTypeCd = a.productTypeCd**

**GROUP BY p.name;**

10.What is the minimum available balance?

**Select min(availbalance) from account;**

11. Produce a list of the total available balance per customer. The balance displayed should be rounded down.

**SELECT custId, FLOOR(SUM(availBalance)) AS total\_available\_balance**

**FROM ACCOUNT**

**GROUP BY custId;**12. The Output lists of EMPLOYEE details in the following formats (only one column should be output):

1. Employees Name; [LAST\_NAME], [FIRST\_NAME] e.g. Smith, Michael

**SELECT lastName || ', ' || firstName AS employee\_name**

**FROM EMPLOYEE;**

b. Employee Position; [FIRST\_NAME] [LAST\_NAME] Position: [TITLE] e.g. Michael Smith Position:

President

**SELECT firstName || ' ' || lastName || ' Position: ' || title AS employee\_position**

**FROM EMPLOYEE;**

13. Consider this text “Fear leads to anger; anger leads to hatred; hatred leads to conflict; conflict leads to suffering”. Write a statement to return the same text but swap the word ‘anger’ for ‘panic buying’.

**Update ‘Fear leads to anger; anger leads to hatred; hatred leads to conflict; conflict leads to suffering’ set ‘Fear leads to panic buying; Panic buying leads to hatred; hatred leads to conflict; conflict leads to suffering.’;**

**’**

14. The data in the CUSTOMER table that holds the FED\_ID is currently held in two different formats. Either nnn-nnnnnn or nn-nnnnnn (where n is a number). The bank wishes to standardise the format so that all values in FED\_ID are stored as nnnnnnnn. Write an update statement to do this.

**UPDATE CUSTOMER**

**SET FED\_ID = CASE**

**WHEN FED\_ID LIKE '\_\_-\_\_\_\_\_\_\_' THEN CONCAT(LEFT(FED\_ID, 1), '0', SUBSTRING(FED\_ID,3))**

**WHEN FED\_ID LIKE '\_-\_\_\_\_\_\_\_' THEN CONCAT('0', LEFT(FED\_ID, 1), SUBSTRING(FED\_ID,3))**

**ELSE FED\_ID**

**END**

**WHERE FED\_ID LIKE '\_\_-\_\_\_\_\_\_\_' OR FED\_ID LIKE '\_-\_\_\_\_\_\_\_';**

15. Write a query to return the year portion of the account transaction date and the number of transactions that took place in each year.

**SELECT txin\_data as year,count(amount) as count(\*)**

**from acc\_transaction**

**group by txin\_data**;

17. Produce a list of customers whose accumulated available balance is less than £5000.

**Select name,sum(aavailbalance) as sum\_avabal**

**from customer**

**where sum\_avabal>=5000;**

18. Produce a report showing the total number of staff assigned to each branch.

**Select sum(branched) as sb, address**

**from branch**

**group by address;**

19. Using only the ACCOUNTS Table; produce a report showing the total number of accounts which have the product codes CHK and SAV.

**WITH AccountProducts AS (**

**SELECT**

**account\_id,**

**COUNT(DISTINCT product\_code) AS product\_count**

**FROM**

**ACCOUNTS**

**WHERE**

**product\_code IN ('CHK', 'SAV')**

**GROUP BY**

**account\_id**

**)**

**SELECT**

**COUNT(\*) AS total\_accounts**

**FROM**

**AccountProducts**

**WHERE**

**product\_count = 2;**