

## Project Component 3: SQL Implementation and Java Database Connectivity (JDBC)

**Dr. Wassim Itani**

**Project Topic:** Design of a database system for a customer loyalty program

**Project Collaboration Model:** This project component should be developed in a **team of 2 members minimum and 4 members maximum**. **You cannot work independently without a team.**

**Due Date/Time:** 11/02/2022 11:59:59 PM.

1. Deliver the .ZIP file (other formats won't be accepted) containing your *insertions.sql* (refer to part 1), *queries.sql* (refer to part 2), *querydisplay.jar* (refer to part 3), and *Readme.txt* files on Blackboard by the above due date/time. The Readme.txt file should contain the name of the group and the names and GMU IDs of the members. Each member should submit the same copy of the ZIP file on Blackboard to secure an entry in the Grade Center. Your .zip file should be named as follows: P3\_[Your Last Name]\_[Your GMUID].zip. For instance, if John Smith with GMUID: G12345678 were to submit this file, John would name it: P3\_Smith\_G12345678.zip

### Project Component 3 Description

The Project Component 3 consists of 3 parts:

#### **Part 1 (25%):**

1. Execute the *LoyaltyFirst\_Impl.sql* script (posted on Blackboard) to generate the LoyaltyFirst database tables.
2. Use the SQL INSERT INTO statement to Insert sample records into the LoyaltyFirst generated database according to the following guidelines:
  - *Insert 20 family records into the Families table. (2%)*
  - *Insert 50 customer records into the Customers table. Assume that all the customers added are part of the LoyaltyFirst program. (3%)*
  - *Insert the customers addresses and phone numbers into the Addresses and Phones tables respectively. At least 5 customers must have 2 addresses and 2 phone numbers. (2%)*
  - *Insert the login information for your customers. You should insert 50 rows corresponding to the number of customers you have in the Customers table (1-to-1 relationship) (1.5%)*
  - *Populate the Point\_Accounts table. It should contain 50 records corresponding to the number of customers you have in the Customers table (Each customer should have an entry in the Point\_Accounts table since all the customers are assumed to be part of the LoyaltyFirst program). (2%)*
  - *Insert data into the Transactions, Products, and Transactions\_Products tables. You should have a minimum of 10 products and 10 transactions (at least 5 different products per transaction). Let some transactions contain some common products with other transactions). (3%)*
  - *Insert the card information into the Cards table. Each customer must have a valid card. Let at least 5 customers have a second expired (non-valid) card. (1.5%)*
  - *Insert the information of at least 10 offers into the Offers table. (1.5%)*

- *Insert the information of 6 branches into the Branches table. (1.5%)*
- *Create an association of your choice between the offers and branches by populating the Offers\_Branches table. (1.5%)*
- *For simplicity, leave the Offers\_Transactions table empty.*
- *Create a set of 5 exchange centers. (1.5%)*
- *Create a list of 15 prizes. (1.5%)*
- *Populate the Redemption\_History table with sample data linking the Customers, Prizes, Point\_Accounts, and Exchg\_Centers tables. You should have a minimum of 20 records in the Redemption\_History table. (2.5%)*

Store the SQL INSERT INTO statements in a file named: ***insertions.sql***. Make sure that your insertions.sql script executes successfully on the LoyaltyFirst database on the GMU Oracle Server.

### **Part 2 (50%):**

Provide the SQL SELECT statements satisfying the following queries. Include your queries in a file named ***queries.sql***. Make sure that your queries.sql script executes successfully on the LoyaltyFirst database on the GMU Oracle Server. **Number your queries using SQL Comments to make it easier on your GTA to grade your project. Not following this guideline would result in a deduction of 10%.**

- 1) *Select the offer id, action, and date provided by a particular branch name. (2.5%)*
- 2) *Display all the transaction references, transaction amounts, the number of points collected from each transaction, and the transaction date for a particular customer name. (2.5%)*
- 3) *Find the branch IDs and the number of offers provided by each branch id. (2.5%)*
- 4) *Find the branch names and the number of offers provided by each branch name. (2.5%)*
- 5) *Display for a particular transaction reference, the transaction reference, date, time, amount, the number of points collected, and the product ids and names included in the transaction with their quantities, prices, and number of points. (2.5%)*
- 6) *Find the number of expired cards available in the database. (2.5%)*
- 7) *Find the customer with the maximum number of expired cards. (2.5%)*
- 8) *Find the redemption history of a particular customer name. You should display the prize ID, prize description, customer name, center id, and number of points redeemed. (3%)*
- 9) *Display the name and occupation of the members of a particular family name. (3%)*
- 10) *Display the sum of points of the members of a particular family ID. (3%)*
- 11) *Display the customer name with the maximum number of collected points. (3%)*
- 12) *Find the total number of points redeemed on a particular date. (3%)*
- 13) *Find the number of prizes redeemed by a particular customer id. (3%)*
- 14) *Find the number of customers who redeemed prizes from a particular center id. (3%)*
- 15) *Find the total number of prizes in the database. (3%)*
- 16) *Display a list of customer names living in Fairfax and whose occupation is Engineer. (2.5%)*
- 17) *Find the list of products not included in any transaction. (3%)*
- 18) *Find the product bought the most by customers. (3%)*

### **Part 3 (25%):**

In this part, the LoyaltyFirst developers ask you to develop a simple standalone application in Java to test the results of Oracle SQL queries. The application consists of 2 main Java Swing JFrames. The first JFrame in Figure 1 contains the connection parameters for accessing the Oracle database server.

Server IP (Host name):

Service Name:

Port Number:

User Name:

Password:

Figure 1: The database connection parameters

Query Text:

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	1987-06-17 ...	AD_PRES	24000			90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1989-09-21 ...	AD_VP	17000		100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	1993-01-13 ...	AD_VP	17000		100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	1990-01-03 ...	IT_PROG	9000		102	60
104	Bruce	Ernst	BERNST	590.423.4568	1991-05-21 ...	IT_PROG	6000		103	60
105	David	Austin	DAUSTIN	590.423.4569	1997-06-25 ...	IT_PROG	4800		103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	1998-02-05 ...	IT_PROG	4800		103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	1999-02-07 ...	IT_PROG	4200		103	60
108	Nancy	Greenberg	NGREENBE	515.124.4569	1994-08-17 ...	PL_MGR	12000		101	100
109	Daniel	Faviet	DFAVIET	515.124.4169	1994-08-16 ...	FL_ACCOUNT	9000		108	100
110	John	Chen	JCHEN	515.124.4269	1997-09-28 ...	FL_ACCOUNT	8200		108	100
111	Ismael	Sciarra	ISCIARRA	515.124.4369	1997-09-30 ...	FL_ACCOUNT	7700		108	100
112	Jose Manuel	Urman	JMURMAN	515.124.4469	1998-03-07 ...	FL_ACCOUNT	7800		108	100
113	Luis	Popp	LPOPP	515.124.4567	1999-12-07 ...	FL_ACCOUNT	6900		108	100
114	Den	Raphaely	DRAPHEAL	515.127.4561	1994-12-07 ...	PU_MAN	11000		100	30
115	Alexander	Khoo	AKHOO	515.127.4562	1995-05-18 ...	PU_CLERK	3100		114	30
116	Shelli	Baida	SBAIDA	515.127.4563	1997-12-24 ...	PU_CLERK	2900		114	30
117	Sigal	Tobias	STOBIAS	515.127.4564	1997-07-24 ...	PU_CLERK	2800		114	30
118	Guy	Himuro	GHIMURO	515.127.4565	1998-11-15 ...	PU_CLERK	2600		114	30

Figure 2: The query execution JFrame

Note that the user can enter any Oracle Server parameters (not necessarily the GMU Oracle Server parameters as demonstrated below). Once the Connect button is clicked and the connection is successful, the application displays the JFrame demonstrated in Figure 2. Very simply, the user can enter any SQL DML query statement (SELECT, DELETE, INSERT INTO, UPDATE) and execute it. If the SQL statement is a SELECT statement, the result is displayed in a JTable component. The JTable should dynamically figure out the number and names of columns in the SQL result and display it on the screen. You may use the ojdbc10.jar JDBC drivers (posted on Blackboard) from Oracle for supporting the database connections. You should package your application in a single Java JAR file containing all the application dependencies including the

JDBC drivers needed for the connection to the database. Name the file ***querydisplay.jar***. The GTAs should be able to execute the command:

*java -jar querydisplay.jar*

to run the application.