**River’s Edge Hotel**

#### Background

The River’s Edge Hotel has decided to automate their reservation and billing systems. You have been hired to design and build the system utilizing the following information.

## Reservation/Registration

Customers can make reservations or can register upon arrival. The following is an example of the form used to collect information about the customer.

|  |  |  |  |
| --- | --- | --- | --- |
| River’s Edge Hotel **Customer Reservation/Registration Card** | | | |
| Customer Last name: | **Smith** | Room Number: | **12** |
| Customer First Name: | **John** | Room Type: | **Two Queen** |
| Address: | **123 Somewhere Street** | Arrival Date: | **23-Aug-2021** |
| City: | **Edmonton** | Departure Date: | **27-Aug-2021** |
| Province: | **AB** | Smoking: | N |
| Postcode: | **T5T 4N6** | Pets: | **N** |
| Credit Card #: | **1234 5678 9012 3456** | Reservation Status: | **P** |

The hotel staff also indicated the following about their processes:

1. Each customer is assigned a unique customer number which is never reused
2. Use of a credit card is optional
3. Each room has a unique room number and is designated as smoking or non-smoking
4. Each room has its own status for availability to pets
5. Each room has its own associated cost and price
6. Each room has a room type and code which indicate the number and sizes of beds in the room as well as whether it is a standard room, kitchenette, suite or president’s suite
7. Each reservation/registration is recorded and assigned a sequential number
8. Staff records the status for each reservation/registration which indicates whether the room is reserved, the customer cancelled, and if the customer paid
9. All registrations are kept for historical purposes
10. Both a reservation status code and a status description are kept on record

## Room Service

While staying at River’s Edge Hotel, a customer can order from room service. The hotel also sells these items to the general public. For a customer staying at the hotel there may be a discount for items ordered from room service. Each item from room service has its own associated cost, selling price, possible discount, and unique item code.

Customer Billing

In addition to the invoice received for the room, customers receive an invoice for any items purchased from room service. Each time a customer makes a purchase from room service an invoice (such as the one below) is produced. A customer may get many of these invoices depending on how many times they purchase items from room service.

In addition, the following information is known about room service invoices:

1. All invoices for a given customer are recorded against that customer’s registration number which allows all bills to be grouped together for payment by the customer at the time of departure
2. The cost and price of all invoice items are kept with the invoice details for historical purposes
3. The total cost and total price of all invoices are kept with the registration for historical purposes
4. Each invoice number is unique
5. All invoice information is kept for historical purposes
6. Each invoice is marked in the system as being paid or not paid

|  |  |  |  |
| --- | --- | --- | --- |
| River’s Edge HotelRoom Service Invoice | | | |
| Customer Last name: | Smith | Room Number: | **12** |
| Customer First Name: | **John** | Room Type: | **D** |
| Registration #: | **179** | Invoice Date: | **24-Aug-2021** |
|  |  | Invoice #: | **8912** |
| Item # | Description | Price | Quantity |
| 528  535  552 | Tea  Omelette  Extra Cheese | 1.50  4.95  0.50 | 1  1  2 |
| Sub Total  GST  Total |  | 7.45  0.37  7.82 |  |

**Part A (Due September 23rd @ 03:00 pm)**

1. Create the following tables and corresponding constraints using only the Create table statement. Ensure that table and column names are **identical** to what is specified below and do **NOT** use double quotes around your table or column names. All constraints must be **explicitly** named, and you must be consistent with your constraint names. **22 Marks**

You can load your tables with the “Data File” which is/will be available from Moodle.

|  |  |  |  |
| --- | --- | --- | --- |
| Customer | Customer\_Code PK | Char (4) | Not Null |
|  | Last\_Name | Varchar2 (20) | Not Null |
|  | First\_Name | Varchar2 (15) | Not Null |
|  | Street\_Address | Varchar2 (40) | Not Null |
|  | City | Varchar2 (35) | Not Null |
|  | Province | Char (2) | Not Null |
|  | Postal\_Code | Varchar2 (7) | Not Null |
|  | Area\_Code | Number (3,0) | Null |
|  | Phone\_Number | Number (10,0) | Not Null |
|  | Credit\_Card\_Number | Varchar2 (16) | Null |
|  | Customer\_Source\_Code FK | Char (1) | Not Null |
|  |  |  |  |
| Customer\_Source | **Customer\_Source\_Code PK** | Char (1) | Not Null |
|  | Customer\_Source\_Description | Varchar2 (30) | Not Null |
|  |  |  |  |
| Invoice | Invoice\_Number PK | Number (6,0) | Not Null |
|  | Registration\_Number FK | Number (6,0) | Not Null |
|  | Invoice\_Date | Date | Not Null |
|  | Paid\_YN | Char (1) | Null |
|  | Invoice\_GST | Number (9,2) | Not Null |
|  | Invoice\_Total | Number (10,2) | Not Null |
|  |  |  |  |
| Invoice\_Item | *Invoice\_Number* *PK FK* | Number (6,0) | Not Null |
|  | ***Item\_Number*** ***PK*** *FK* | Number (4,0) | Not Null |
|  | Quantity\_Sold | Number (5,0) | Not Null |
|  | Selling\_Cost | Number (5,2) | Not Null |
|  | Selling\_Price | Number (5,2) | Not Null |
|  |  |  |  |
| Item | Item\_Number PK | Number (4,0) | Not Null |
|  | Item\_Description | Varchar2 (45) | Not Null |
|  | Current\_Cost | Number (5,2) | Null |
|  | Current\_Price | Number (5,2) | Null |
|  | Discount | Number (3,0) | Null |
|  |  |  |  |
| Registration\_Status | Registration\_Status\_Code PK | Char (1) | Not Null |
|  | Registration\_Status\_Descriptor | Varchar2 (20) | Not Null |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Registration | Registration\_Number PK | Number (6,0) | Not Null |
|  | Room\_Number FK | Number (3,0) | Not Null |
|  | Arrival\_Date | Date | Not Null |
|  | Departure\_Date | Date | Not Null |
|  | Customer\_Code FK | Char (4) | Not Null |
|  | *Registration\_Status\_Code* *FK* | Char (1) | Not Null |
|  | Total\_Cost | Number (12,2) | Null |
|  | Total\_Price | Number (12,2) | Null |
|  |  |  |  |
| Room | Room\_Number PK | Number (3,0) | Not Null |
|  | Smoking\_SN | Char (1) | Not Null |
|  | Pet\_Status | Char (1) | Not Null |
|  | Room\_Rate | Number (6,2) | Not Null |
|  | *Room\_Type\_Code FK* | Char (1) | Not Null |
|  |  |  |  |
| Room\_Type | Room\_Type\_Code PK | Char (1) | Not Null |
|  | Room\_Type\_Description | Varchar2 (35) | Not Null |

Additional table information:

* Primary keys are **bolded** and are denoted by PK
* Foreign keys are *italicized* and are denoted by FK
* All monetary values must be greater than or equal to 0 and default to zero
* The Quantity\_Sold field must be greater than zero and default to 1
* Item discount must be between 0 and 100
* All date fields must default to today’s date and be greater than January 01, 2018
* The departure date must not be prior to the arrival date
* The Paid\_YN field will accept the values Y or N and will default to N

1. Each room is designated as Smoking (S) or Non-smoking (N) and will default to N

* Each room has a pet status (N – No Pets, D – Dogs Only, C – Cats Only, A – All Pets) and will default to A
* The province must be two upper case alpha characters and default to Alberta (AB)
* The customer code format is AAA9 where A is any upper case alpha character and 9 is a number between 0 and 9
* Null and Not Null are constraints and as such must be named
* While Null is currently the default, you are not guaranteed this will not change; therefore you must explicitly declare the Null constraint where indicated

1. Create the following sequences in your database. Ensure that the sequence names are **identical** to what is specified below and additional numbers are not pre-allocated. **1 mark**

|  |  |  |
| --- | --- | --- |
| Sequence Name | Initial Value | Increment By |
| Seq\_Registration | 50 | 1 |
| Seq\_Invoice | 12100 | 10 |
| Seq\_Item | 750 | 1 |
| Seq\_Room | 35 | 5 |

**Part A Lab Submission is to include the following:**

* Ensure your tables and sequences are created within the Oracle DMIT environment (not just on your laptop or home computer)
  + If you have trouble with a particular constraint, **please** contact your instructor
  + Please do not change anything with respect to your tables and/or sequences after the due date/time until your lab has been **marked and returned**
* An emailed discussion about part A of the lab including:
  + How long the lab took to complete (split between in class and out of class time)
  + Any significant problems that were encountered
  + Whether or not there was enough material provided in the lectures to complete the lab
  + What, if any, changes, or improvements could be made to the lab
* One electronic non-compressed **text** file named Lab1A\_YourLastName\_YourFirstName.sql will be submitted to Moodle
* If you need to re-submit your lab after you have submitted but prior to the due date/time, email your instructor a request to adjust your submission to Draft Mode, then you can submit the revised file
* Do not include your password, the lab specification or the “Data File” in your lab submission
* Any additional requirements as specified by your instructor

**Part B (Due October 21st @ 03:00 pm)**

1. Write a procedure called **PR\_Q3** that will populate the **Area\_Code** field. If the postal code is for northern Alberta (i.e. T5 to T9), set the area code to 780; for southern Alberta (i.e. T0 to T4), set the **Area\_Code** to 403; for non-Alberta addresses, set the **Area\_Code** to 555. To facilitate the “other” Alberta area codes, for all Alberta residents where the last digit of their postal code is between 6 and 9, set the **Area\_Code** to 587; for all Alberta residents where the last digit of their postal code is between 0 and 3, set the **Area\_Code** to 825. Also update the **Phone\_Number** field to include the area code (i.e. 4302211 with a postal code of T6N 5T9 would get updated to 5874302211). Use at least one cursor in your solution.

For testing purposes, please add "authid current\_user" to your procedure.

Note: the “new” Alberta area code checks (i.e. last digit between 0 and 3 or 6 and 9) take precedence over the 2nd digit (i.e. southern Alberta 0 to 4 and northern Alberta 5 to 9) checks.

**Procedure code 6 marks, Test case 2 marks**

Your test case will consist of three steps:

1. A select **and** display of the **appropriate** data from the **appropriate** table(s) **before** the procedure is executed
2. Execute your procedure (ensure your execute statement is included in your lab submission)
3. A select **and** display of the **appropriate** data from the **appropriate** table(s) **after** the procedure is executed
4. Create an overloaded function **(FN\_Q4)** within a package **(PKG\_Q4)** that returns customer details as a string when passed a customer’s first and last names (in that order) or a customer code. If the customer does not exist or if the first and last name combination is not unique, or if the customer has never stayed, return appropriate error messages. If the customer has **stayed** (i.e. did not cancel or was a no show) only once, return a string displaying the room number, pet status description, customer source description and total charges for all invoices. If the customer has **stayed** more than once, return a string displaying the number of visits, the average stay in days, the total of all room service invoices and the total of all room charge invoices. The customer code, first and last names (if available) must be part of all output strings. Input case of the parameters is not guaranteed. Use exception handling in your solution. All customers are guaranteed to be in the registration table at least once. Hint: the status of a customer who has **stayed** is determined by the registration status code / descriptor.

For testing purposes, please add "authid current\_user" to your **package specification only**.

**Note:** Each function must stand alone within the package (i.e. one function must NOT call the other function). **Package code 10 marks, Test cases 4 marks**

Test cases: demonstrate the functionality of your code with the following scenarios:

* Non-unique customer name combination
* Non-existent customer (names and code)
* Existing customer (names and code) who has not stayed
* Existing customer (names and code) who has stayed once
* Existing customer (names and code) who has stayed more than once

Each test case will consist of two steps:

1. A select **and** display of the **appropriate** data from the **appropriate** table(s)
2. Call your function (ensure your call statement is included in your lab submission) **and** display the results
3. Create a trigger called **TR\_Q5,** which will not allow records to be inserted to, updated or deleted from, the invoice item table if the corresponding invoice has been paid. If an error occurs, stop processing and display an appropriate error. **Trigger code 6 marks, Test cases 3 marks**

Test Cases: Test cases displaying **both** successful **and** unsuccessful completions for **all** three DML commands [insert, update, delete] are **required**.

Each test case will consist of three steps:

1. A select **and** display of the **appropriate** data from the **appropriate** table(s) **before** the code is executed
2. Execute your DML statement (ensure your DML statement is included in your lab submission)
3. The resulting message (success or failure) **and** a select **and** display of the **appropriate** data from the **appropriate** table(s) **after** the DML statement is executed

**Part B Lab Submission is to include the following:**

* Ensure your procedure, package and trigger are created within the Oracle DMIT environment (not just on your laptop or home computer)
  + If you have trouble with a particular question, **please** contact your instructor
  + Please do not change anything with respect to your procedure, package, and/or trigger after the due date/time until your lab has been **marked and returned**
* A short, emailed discussion about the lab including:
  + What you liked and/or disliked about the lab
  + How long the lab took to complete (split between in class and out of class time)
  + Any significant problems that were encountered
  + Whether or not there was enough material provided in the lectures to complete the lab
  + What, if any, changes, or improvements could be made to the lab
* Two electronic files will be submitted to Moodle:
  + Your source code as a single **text** file named Lab1B\_Source\_LastName\_FirstName.sql
  + Your test cases as a single **text** file named Lab1B\_TestCases\_LastName\_FirstName.sql
  + Compressed files (i.e. zip, rar, etc.) or folders will NOT be accepted
* If you need to re-submit your lab after you have submitted but prior to the due date/time, email your instructor a request to adjust your submission to Draft Mode, then you can submit the revised files
* Ensure there are no rollback, or commit, statements in your code!
* Do not include your password, the lab specification or the “Data File” in your lab submission
* **Select \* is NOT acceptable anywhere in your lab (including test cases and/or commented out code)**
* Any additional requirements as specified by your instructor

***Lab Expectations:***

* All code **MUST** be executable within the Oracle DMIT environment
* Tables, columns, sequences, procedures, functions, packages, and triggers **MUST** be named exactly as indicated
* All cursors must be explicitly declared, opened, retrieved from, and closed
* All code will be documented, where appropriate, and will follow a convention such as standards document available on Moodle
* Any additional requirements as specified by your instructor