CSCI 6950 – Indexing and Query Optimization

For the following tables and SQL statement, select indexes that balance retrieval and update requirements. For each table, justify your choice using the rules. If you don’t have the database in Oracle Express, you can upload it again using the sql script attached to this assignment.

**Customer(**CustNo, CustFirstName, CustLastName, CustCity, CustState, CustZip, CustBal)

**OrderTbl(**OrdNo, OrdDate, *CustNo*)

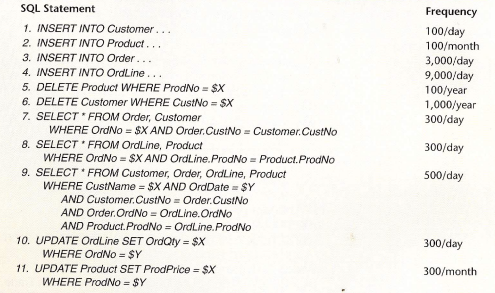
FOREIGN KEY CustNo REFERENCES Customer

**OrdLine**(OrdNo, ProDNo, OrdQty)

FOREIGN KEY OrdNo REFERENCES OrderTbl

FOREIGN KEY ProdNo REFERENCES Product

**Product**(ProdNo, ProdName, ProdMFG, ProdColor, ProdPrice)



1. For the four tables Customer, OrderTbl, OrdLine and Product identify what columns are going to be automatically indexed in ORACLE (primary keys and unique columns)? Check your answer using SQL Developer.

Ans : Customer.CustNo

OrderTbl.OrderNo

Product.ProdNo

Orderline.OrdNo and ProdNo

Graphical user interface, text, application, email

Description automatically generated

1. For the four tables Customer, OrderTbl, OrdLine and Product identify what columns are good choices for indexing (using a B-Tree or a Bitmap index)?

Ans: B-Tree will be good choices for indexing.

Customer table – an index on Customer.Last\_Name

OrderTbl – an index on OrderTbl.OrdDate

Product – an index on Product.ProdMFG , Product.ProdName

1. For query 9 above, list the possible join orders considered by the query optimization component.

Ans: 1. (((Customer, Order), OrdLine), Product)

2. (((Order, OrdLine), Customer), Product)

3. (((Order, OrdLine), Product), Customer)

4. (((OrdLine, Product), Order), Customer)

5. ((Customer, Order), (OrdLine, Product))

6. ((OrdLine, Product), (Customer, Order))

1. In SQL Developer write CREATE INDEX statements to create all indexes from exercise 2.

Ans: CREATE INDEX ProdNoIndex ON Product (ProdNo) ;

CREATE UNIQUE INDEX ProdNoIndex ON Product (ProdNo) ;

CREATE BITMAP INDEX ProdNoIndex ON Product (ProdNo);

1. Write Drop Index statements for the indexes created in exercise 3.

Ans: DROP INDEX OrdNo ON OrderTbl;

1. Create two bitmap indexes, one for the CustState Column in table Customer and one for ProdMFG in table Product. Write two queries that use the where clause with a condition based on columns CustState and ProdMFG. Run the two queries twice, once without an index, once with a BTree index, and once with a Bitmap index. Record the running times in the following table:

|  |  |  |
| --- | --- | --- |
|  |  | Time in seconds |
| Customer Table | No index | 0.286 |
| Customer Table | Bitmap index | Error |
| Customer Table | BTree index | Error |
| Product Table | No index | 0.287 |
| Product Table | Bitmap index | Error |
| Product Table | BTree index | Error |

Query Product

CREATE BITMAP INDEX index\_ProdMFG ON CR\_Product(ProdMFG);

CREATE INDEX idx\_ProdMFG ON CR\_Product(ProdMFG);

Query Customer

CREATE BITMAP INDEX index\_CustState ON CR\_Customer(CustState);

CREATE INDEX idx\_CustState ON CR\_Customer(CustState);

1. The following query list the last name, and city for each customer who has a balance greater that $150 and placed an order in February 2007.

SELECT CustNo, CustFirstName, CustLastName, CustCity

FROM Customer

WHERE CustBal > 150 AND EXISTS

( SELECT CustNo

FROM OrderTbl

WHERE OrdDate BETWEEN '1-Feb-2007' AND '28-Feb-2007'

AND Customer.CustNo = OrderTbl.CustNo );

SELECT CustNo, CustFirstName, CustLastName, CustCity

FROM cr\_Customer

WHERE CustBal > 150 AND EXISTS

( SELECT CustNo

FROM cr\_OrderTbl

WHERE OrdDate Like '%Feb%'

AND cr\_Customer.CustNo = cr\_OrderTbl.CustNo );

Run the query in SQL developer and retrieve the Explain Plan. Export it as html. Change the extension of the file to .html and open it in a browser. Here you can select and copy. Paste the explain table below.

Table 1 query

CUSTNO CUSTFIRSTNAME CUSTLASTNAME CUSTCITY

-------- -------------------- ------------------------------ ------------------------------

C9857432 Homer Wells Seattle

Query.html

Graphical user interface, text, application, Teams

Description automatically generated

The explanation of the above screenshot:

Graphical user interface

Description automatically generated

|  |
| --- |
|  |

Which select statement has lower cost? Why?

Ans: Both of Statement has same outcome. Since the OrderDate is same in both statement.

1. Indexes on combinations of columns are not useful as indexes on individual columns. Consider a combination index on two columns, CustState and CustCity, where CustState is the primary ordering and CustCity is the secondary ordering. For what kinds of conditions can the index be used? For what kinds of conditions is the index not useful?

Ans: In small or sparsely populated tables, indexes should not be used. Point to notice that low sparsity and small tables are not the same thing. Even yet, a search condition in a sparsely sparse table can return a large portion of the table rows, making the index operation unaffordable and the full table scan an option. Using the same strategy, stay away from making indexes for tables with few rows and few characteristics unless you absolutely need to verify that each field has only unique values.