**Homework Week 5**

**Chapter 5**

**Answers to Problems and Exercises**

1. StudentID in STUDENT because it is a primary key and the index would enforce uniqueness of the key; also, StudentID in STUDENT and in REGISTRATION is used in a WHERE clause for joining the STUDENT and REGISTRATION tables, so it likely makes sense to create an index on StudentID in REGISTRATION as well.

GPA in STUDENT because it is a nonkey cluster attribute used to qualify record retrieval

StudentName in STUDENT because it is a nonkey attribute used to sort records

StudentID, CourseID in REGISTRATION because it is a concatenated primary key and the index would enforce uniqueness of the key

1. CREATE UNIQUE INDEX STUPKINDX ON STUDENT (StudentID);

CREATE INDEX STUDREGINDX ON REGISTRATION (StudentID);

CREATE INDEX CLUST\_INDX

ON STUDENT (GPA)

CLUSTER;

CREATE INDEX NAMEINDX ON STUDENT (StudentName);

CREATE UNIQUE INDEX REGSPKINDX

ON REGISTRATION (StudentID, CourseID);

1. How will numeric value 3,456,349.2334 be stored assuming various Oracle data types?

|  |  |  |
| --- | --- | --- |
|  | Oracle data type | Stored Value |
| a. | NUMBER(11) | 3456349 |
| b. | NUMBER(11,1) | 3456349.2 |
| c. | NUMBER(11,-2) | 3456300 |
| d. | NUMBER(6) | Not accepted; exceeds precision |
| e. | NUMBER | 3456349.2334 |

1. Recommendations for denormalization:

EMPLOYEE SCHEDULE

(DepartmentID, EmployeeID, WhereWork, EmployeeName, EmployeeAddress, Date)

*A many-to-many relationship (associative entity) with nonkey attributes*: Rather than joining three files to extract data from the two basic entities in the relationship, it may be advisable to combine attributes from one of the entities into the record representing the relation in the many-to-many relationship, thus avoiding one join in many data access modules. This approach is advantageous as this joining will occur frequently.

DEPARTMENT

(DepartmentID, ManagerID, SalesGoal, StoreID, Region, ManagerID, SquareFeet)

This *reference data* denormalization option wouldn't be recommended since the table STORE is further related to a table MANAGER, and there are probably more than just a few departments in each STORE.

9.. Recommendations for denormalization:

One possibility for denormalization would be the inclusion of SpecialtyDescription attribute in the PLAYER relation. It appears that the PlayerSpecialtyCode refers to the SpecialtyCode in the SPECIALTY relation, using SPECIALTY as reference information for Players. To make the right decision, you would need to know more about Player specialties, in terms of occurrence (can they have more than one, in the future?) and the expected size of the SpecialtyDescription attribute. Denormalizing the relations by adding the SpecialtyDescription data to the PLAYER relation might be adding a lot of redundant information that could take up a great deal of storage.

There might be opportunities for further denormalization in these relations, but the following questions need to be asked before proceeding with any further denormalization:

* What is the meaning and the contents of the TeamLocation attribute in the TEAM relation? Is this a code used to reference the LocationID in the LOCATION relation? Or is this a character field storing different information about the Team’s location? Can a TEAM instance have more than one location?
* What is the meaning and the contents of the ManagerTeam attribute in the MANAGER relation? Does this attribute refer to the TeamID attribute in the TEAM relation? Is a Manager instance also a Player instance?

13. A row selection qualification clause will be used:

WHERE (Major = “MIS“ or Major = “Computer Science“) And Age > 25 And MaritalStatus = “single“) Or (Major = “Computer Engineering“ And MaritalStatus = “single“ And HomeZipcode = 45462). Indexes on these fields can be used to considerable advantage in this situation. Assume that each index qualification (e.g., Major = “MIS“) produces a list of record numbers for the records satisfying that qualification. Lists can be merged to process OR operators, and lists can be intersected to process AND operators. Indexes may be scanned in main memory, and the list operations also done without accessing secondary memory, thus composing the list of qualified records very quickly. Only then does secondary memory need to be accessed for only those records that satisfy the whole query.

16. A cluster is defined by the tables and the column or columns by which the tables are usually joined. The column by which they are joined (foreign key) would need to have the same value in the two tables for the adjacent records. If the tables are populated with data before clustering occurs, this is much harder to achieve. Hence, in Oracle, tables are assigned to a cluster at the time of their creation.

1. In general, the answer depends on the number of processors available. For the general structure mentioned in the problem, each set of conditions within parentheses (called a conjunction) could be given to a separate processor, then the results from each processor would be intersected to obtain the final result. If more processors are available, then each condition within a conjunction could be assigned to its own processor.

As an illustration of what might typically happen, consider the qualification clause from Problem and Exercise 13:

SELECT StudentID, StudentName

FROM STUDENT

WHERE (Major = “MIS“ or Major = “Computer Science“) And Age > 25 And

MaritalStatus = “single“ Or (Major = “Computer Engineering“ And MaritalStatus = “single“ And HomeZipcode = 45462);

There are two general approaches for parallel processing of a query:

1. To ensure that subsequent scans of this table are performed in parallel using at least three processors, you would first alter the structure of the table with the SQL command:

ALTER TABLE STUDENT PARALLEL 3,

and then run the query itself.

1. The second option would be to give the DBMS a hint within the query. This will force it to process the query in a certain way. In Oracle:

SELECT /\*+ FULL(STUDENT) PARALLEL(STUDENT,3) \*/COUNT(\*)

FROM STUDENT

WHERE (Major = “MIS“ or Major = “Computer Science“) And Age > 25

And MaritalStatus = “single“ Or (Major = “Computer Engineering“ And MaritalStatus = “single“ And HomeZipcode = 45462);

would enforce a full scan on the table STUDENT, and its processing in parallel, by three CPUs. **Note:** In Oracle, parallel processing is possible only when a table is scanned, not when it is accessed through an index.