BACHELOR OF COMPUTER APPLICATIONS

SEMESTER 3

DCA2102

DATABASE MANAGEMENT SYSTEM

Unit 5

An Introduction to RDBMS

Table of Contents

SL No	Topic	Fig No / Table / Graph	SAQ / Activity	Page No
1	Introduction			3
	1.1 Objectives			
2	An Informal look at the Relational Model	1, 2	1	4 - 5
3	Relational Database Management System	3, 4, 5	2	6 – 9
4	RDBMS Properties	6, 7	3	10 - 14
	4.1 The Entity-Relationship Model			
5	Overview of Relational Query Optimization		4	15
6	System Catalog in a Relational DBMS		5	16 - 18
	6.1 Information Stored in the System Catalog			
	6.2 How Catalogs are Stored	8		
7	Summary			19
8	Terminal Questions			19
9	Answers			20 - 21

1. INTRODUCTION

A **relational database** is a collection of relations with distinct relation names. The **relational database schema** is the collection of schemas for the relations in the database. For example, a university database contains schemas for relations called Students, Faculty, Courses, Rooms, Enrolled Numbers, Teachers, etc. An **instance** of a relational database is a collection of relation instances, one per relation schema in the database schema; of course, each relation instance must satisfy the domain constraints in its schema.

1.1 Objectives:

By the end of the Unit 5, the learners should be able to understand:

- *The relation, relation schema, and domain name concept*
- *The relational database components like tables, columns, etc.*
- *Properties of RDBMS including entity-relationship models*
- *Optimization in RDBMS*
- *System catalogs in RDBMS*

2. AN INFORMAL LOOK AT THE RELATIONAL MODEL

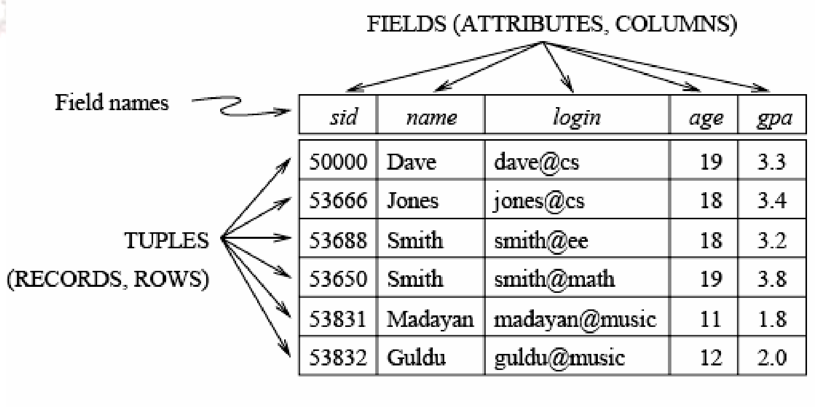
The main construct for representing data in the relational model is a **relation**. A relation consists of a **relation schema** and a **relation instance**. The relation instance is a table, and the relation schema describes the column heads for the table. We first describe the relation schema and then the relation instance. The schema specifies the relation's name, the name ofeach **field** (or **column**, or **attribute**), and the **domain** of each field. A domain is referred to in a relation schema by the **domain name** and has a set of associated **values**.

We use the example of student information in a university database to illustrate the parts of a relation schema:

Students(sid: string, name: string, login: string, age: integer, gpa: real)

This says, for instance, that the field name *sid* has a domain name string. The set of values associated with the domain string is the set of all character strings. We now turn to the instances of a relation. An **instance** of a relationis a set of **tuples**, also called **records**, in which each tuple has the same number of fields as the relation schema. A relation instance can be thought of as a *table* in which each tuple is a *row*, and all rows have the same number of fields. (The term *relation instance* is often abbreviated to just *relation* when there is no confusion with other aspects of a relation such as its schema.)

An instance of the Students relation appears in Figure 5.1. The instance *S*1 contains six tuples and has, as we expect from the schema, five fields. Notethat no two rows are identical.


FIELDS (ATTRIBUTES, COLUMNS)
Field names
TUPLES
(RECORDS, ROWS)

sid	*name*	*login*	*age*	*gpa*
50000	Dave	dave@cs	19	3.3
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@math	19	3.8
53831	Madayan	madayan@music	11	1.8
53832	Guldu	guldu@music	12	2.0

Fig 5.1: An Instance *S*1 of the Student's Relation

This is a requirement of the relational model - each relation is defined to bea *set* of unique tuples or rows. The order in which the rows are listed is not important.

sid	*name*	*login*	*age*	*gpa*
53831	Madayan	madayan@music	11	1.8
53832	Guldu	guldu@music	12	2.0
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@math	19	3.8
53666	Jones	jones@cs	18	3.4
50000	Dave	dave@cs	19	3.3

Fig 5.2: An Alternative Representation of Instance *S*1 of Students

Figure 5.2 shows the same relation instance. If the fields are named, as in our schema definitions and figures depicting relation instances, the order of fields does not matter either. However, an alternative convention is to list fields in a specific order and to refer to a field by its position. Thus ***sid*** is field 1 of Students, *login* is field 3, and so on. If this convention is used, the order of fields is significant. Most database systems use a combination of these conventions. For example, in SQL the named fields' convention is used in statements that retrieve tuples, and the ordered fields' convention is commonly used when inserting tuples.

A relation schema specifies the domain of each field or column in therelation instance.

These **domain constraints** in the schema specify an important condition that we want each instance of the relation to satisfy: The values that appear in a column must be drawn from the domain associated with that column. Thus, the domain of a field is essentially the *type* of that field, in programming language terms, and restricts the values that can appear inthe field. Domain constraints are so fundamental in the relational model that we will henceforth consider only relation instances that satisfy them; therefore, relation instance means relation instance that satisfies the domainconstraints in the relation schema.

Self-Assessment Questions – 1

1. A relation consists of a___________and a relation instance.
2. An instance of a relation is a set of tuples, also called___________.
3. Relation instance means relation instance that satisfies the___________ in the relation schema.

3. RELATIONAL DATABASE MANAGEMENT SYSTEM

A Relational Database Management System (RDBMS) provides a comprehensive and integrated approach to information management. A relational model provides the basis for a relational database. A relational model has three aspects:

1. Structures
2. Operations
3. Integrity rules

Structures consist of a collection of objects or relations that store data. An example of a relation is a table. You can store information in a table and use the table to retrieve and modify data.

Operations are used to manipulate data and structures in a database. Whenusing operations, you must adhere to a predefined set of integrity rules.

Integrity *rules* are laws that govern the operations allowed on data in a database. This ensures data accuracy and consistency.

Relational database components as shown in Figure 5.3 include:

- Table
- Row
- Column
- Field
- Primary key
- Foreign key

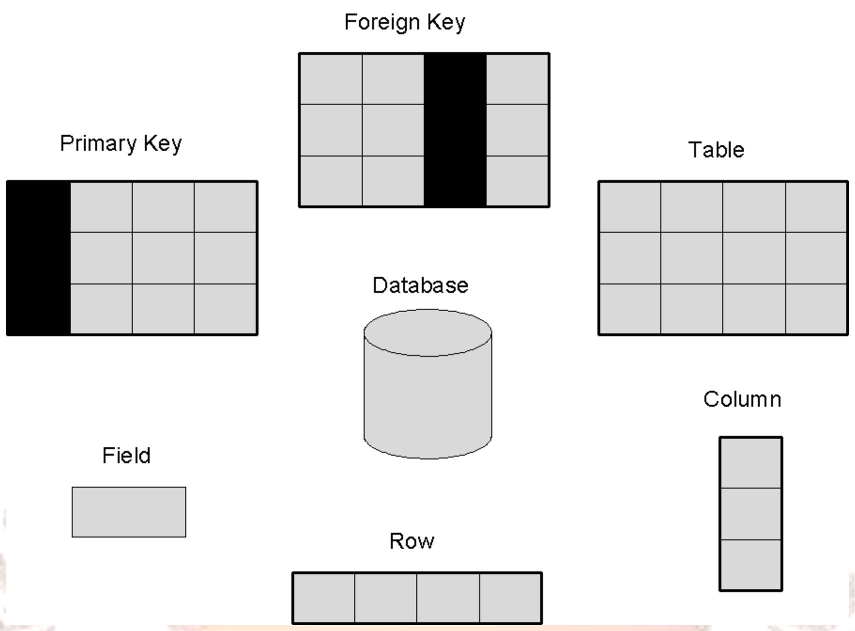


Fig 5.3: Relational database components

A Table is a basic storage structure of an RDBMS and consists of columns and rows as shown in Figure 5.4. A table represents an entity. For example,the S_DEPT table stores information about the departments of an organization.

A Row is a combination of column values in a table and is identified by a primary key. Rows are also known as records. For example, a row in the table S_DEPT contains information about one department.

A Column is a collection of one type of data in a table. Columns represent the attributes of an object. Each column has a column name and contains values that are bound by the same type and size. For example, a column in the table S_DEPT specifies the names of the departments in the organization.

A Field is an intersection of a row and a column. A field contains one data value. If there is no data in the field, the field is said to contain a NULLvalue.

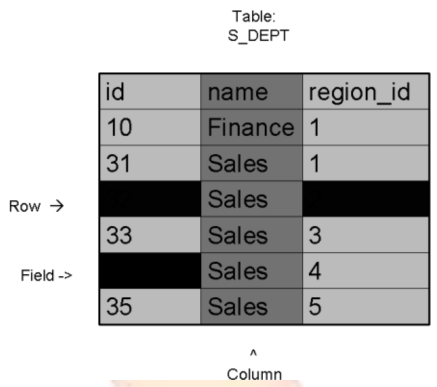


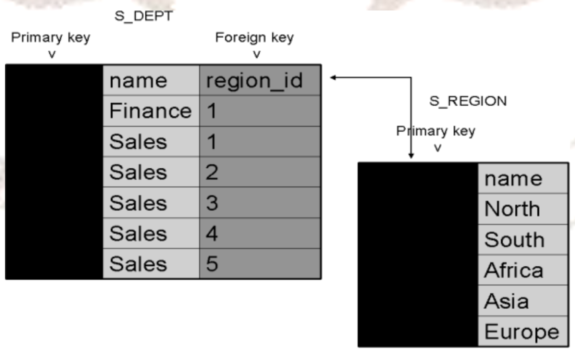
Table:
S_DEPT
Row →
Field ->
^
Column

id	name	region_id
10	Finance	1
31	Sales	1
	Sales	
33	Sales	3
	Sales	4
35	Sales	5

Fig 5.4: Table, Row, Column & Field

A **Primary key** is a column or a combination of columns that is used to uniquely identify each row in a table. For example, the column containing department numbers in the S_DEPT table is created as a primary key, and therefore every department number is different. A primary key must contain a value. It cannot contain a NULL value.

A Foreign key is a column or set of columns that refers to a primary key of another table. You use foreign keys to establish principle connections between, or within, tables. A foreign key must either match a primary key or else be NULL. Rows are connected logically when required. The logical connections are based upon conditions that define a relationship between corresponding values, typically between a primary key and a matching foreign key. This relational method of linking provides great flexibility as it is independent of physical links between records. The primary and foreign key concept is shown in Figure 5.5.



	name	region_id
	Finance	1
	Sales	1
	Sales	2
	Sales	3
	Sales	4
	Sales	5

	name
	North
	South
	Africa
	Asia
	Europe

Fig 5.5: Primary & Foreign key

Self-Assessment Questions – 2

4. ___________consist of a collection of objects or relations that store data.
5. Operations are used to_________data and structures in a database.
6. Integrity Rules ensures data accuracy and_____.
7. A_______is a basic storage structure of an RDBMS and consists ofcolumns and rows.
8. A Row is a combination of column values in a table and is identified by a _________key.
9. A_________is an intersection of a row and a column.

4. RDBMS PROPERTIES

An RDBMS is easily accessible. You execute commands in the Structured Query Language (SQL) to manipulate data. SQL is the International Standards Organization (ISO) standard language for interacting with an RDBMS. The interaction between the SQL and database is shown in Figure 5.6.

An RDBMS provides full data independence. The organization of the data isindependent of the applications that use it. You do not need to specify the access routes to tables or know how data is physically arranged in a database.

A relational database is a collection of individual, named objects. The basic unit of data storage in a relational database is called a table. A tableconsists of rows and columns used to store values. For access purposes, the order of rows and columns is insignificant. You can control the access order as required.

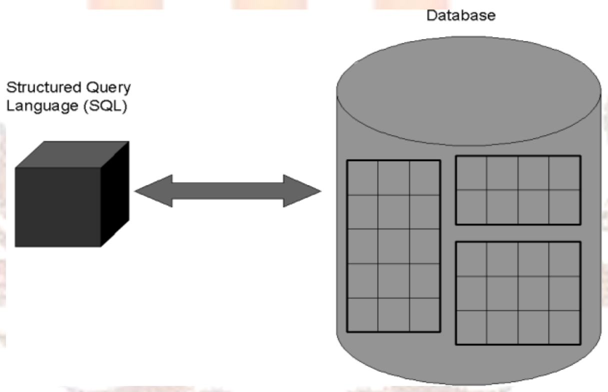
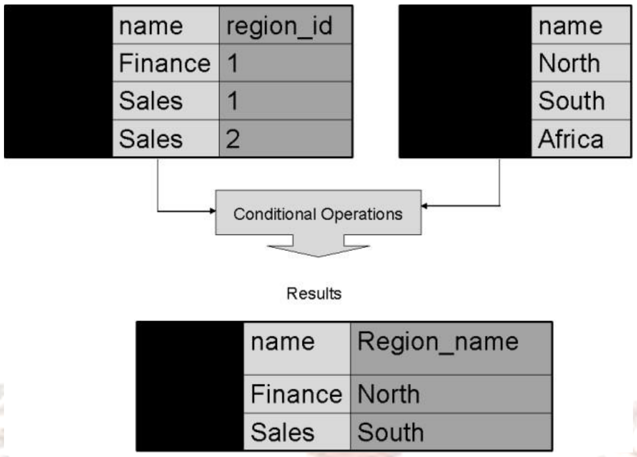


Fig 5.6: SQL & Database

When querying the database, you use conditional operations such as joins and restrictions. A join combines data from separate database rows. A restriction limits the specific rows returned by a query as shown in Figure 5.7. We can learn more details about join operations under relational algebra in Unit 7.



name	region_id
Finance	1
Sales	1
Sales	2

name
North
South
Africa

name	Region_name
Finance	North
Sales	South

Fig 5.7: Conditional operations

An RDBMS enables data sharing between users. At the same time, you can ensure the consistency of data across multiple tables by using integrity constraints. An RDBMS uses various types of data integrity constraints. These types include an entity, column, referential and user-defined constraints.

The constraint, entity, ensure the uniqueness of rows, and the constraint column ensures consistency of the type of data within a column. The other type, referential, ensures the validity of foreign keys, and user-definedconstraints are used to enforce specific business rules.

An RDBMS minimizes the redundancy of data. This means that similar data is not repeated in multiple tables.

4.1 The Entity-Relationship Model

The entity-relationship model is a tool for analyzing the semantic features of an application that are independent of events. This approach includes a graphical notation, which depicts entity classes as rectangles, relationships as diamonds, and attributes as circles or ovals. For a complex situation, a partial entity-relationship diagram may be used to present a summary of the entities and relationships that do not include the details of the attributes.

The entity-relationship diagram provides a convenient method for visualizing the interrelationships among entities in a given application. This tool has proven to be useful in making the transition from an information application description to a formal database schema. The entity-relationship model is used for describing the conceptual schema of an enterprise, without attention to the efficiency of the physical database design. The entity-relationship diagrams are then turned into a logical schema in which the database is implemented.

Short definitions of some of the basic terms that are used for describing important entity-relationship concepts are:

1. **Entity:** An entity is a thing that exists and is distinguishable.
   a) Entity instance. An instance is a particular occurrence of an entity. For example, each person is an instance of an entity Person, each car is an instance of an entity Car, etc.
   b) Entity class. A group of similar entities is called an entity class or entity type. An entity class has common attributes.

2. **Attributes:** Attributes describe the properties of entities and relationships.
   a) **Simple and composite attributes:** A simple attribute is the smallestsemantic unit of data, which is atomic (no internal structure). Acomposite attribute can be subdivided into parts, e.g., address (street, city, state, zip).
   b) **Single and multivalued attributes:** Single attributes have a single value for a particular entity. Multivalued attributes have multiple values of an attribute for a particular entity; e.g., degrees or courses that a student can have or take.
   c) **Domain:** Conceptual definition of attributes: a named set of scalar values, all of the same type, and a pool of possible values.

3. **Relationships:** A relationship is a connection between entities. For example, a relationship between PERSONS and AUTOMOBILES could be an "OWNS" relationship. That is to say, automobiles are owned by people.
   - **Is-a hierarchies:** A special type of relationship that allows attribute inheritance. For example, to say that a truck is an automobile and an automobile has a model and serial number implies that a truck also has a model and serial number.

4. **Keys:** The key uniquely differentiates one entity instance from all others in the entity. A key is an identifier.

a) **Primary Key:** Identifier used to uniquely identify one particularinstance of an entity.

A primary key

- can be one or more attributes (e.g., consider substituting a single concatenated key attribute for multiple attribute key)
- must be unique within the domain (not just the current data set),
- its value should not change over time,
- must always have a value, and
- is created when no obvious attribute exists. Each instance hasa value.

b) **Candidate Key**

When multiple possible identifiers exist, each of them is acandidate key.

c) **Concatenated Key.**

Key is made up of parts which, when combined, become a uniqueidentifier. Multiple attribute keys are concatenated keys.

d) **Borrowed Key Attributes.**

If an is-a relationship exists, the key of the more general entity is also a key of the sub-entities. For example, if the serial number is a keyfor automobiles, it would also be a key for trucks.

e) **Foreign Keys.** Foreign keys reference a related table through the primary key of that related table.

An ER schema may identify certain constraints to which the content of the data must conform.

Two of the most important types of constraints are:

1. The mapping cardinality of a relationship indicates the number of instances in entity E1 that can or must be associated with instances in entity E2:
   a) One-One Relationship. For each entity instance in one entity there isat most one associated entity instance in the other entity. For example, for each husband, there is at the most one current legal wife. A wife has at the most one current legal husband.

b) Many-One Relationships. One entity instance in entity E2 is associated with zero or more entity instances in entity E1, but each entity instance in E1 is associated with at most one entity instance inE2. For example, a woman may have many children but a child has only one birth mother.

c) Many-Many Relationships There are no restrictions on how many entity instances in either entity are associated with a single entity instance in the other. An example of a many-to-many relationship would be students taking classes. Each student takes many classes.Each class has many students.

Mapping cardinality is derived from cardinality constraints. The cardinality constraint between two entities E1 and E2, denoted by (m,n), specifies that an instance in E1 appears in E2 at least m andat most n times. Mapping cardinality takes the maximum number of the cardinality constraint for each entity in a relationship.

2. Existence dependence. If the existence of an entity instance x depends on the existence of an entity instance y, then x is said to be existence dependent on y. If y is deleted so is x. For example – loan_payment is existence dependent on loan_number. If loan_number is deleted so is loan_payment.

Self-Assessment Questions – 3

10. ___________ is the International Standards Organization (ISO) standard language for interacting with an RDBMS.
11. The basic unit of____________in a relational database is called a table.
12. A____________combines data from separate database rows.
13. An RDBMS enables data____________between users.
14. An RDBMS minimizes the__________of data.
15. A degree of a student is__________valued attribute.
16. The__________uniquely differentiates one entity instance from all others inthe entity.
17. A Concatenated Key is made up of parts which, when combined,become a ____________identifier.

5. OVERVIEW OF RELATIONAL QUERY OPTIMIZATION

The goal of a query optimizer is to find a good evaluation plan for a given query. The space of plans considered by a typical relational query optimizer can be understood by recognizing that *a* query is essentially treated as a σ-π-x algebra expression, with the remaining operations (if any, in a given query) carried out on the result of the σ-π-x expression. Optimizing such a relational algebra expression involves two basic steps:

a) Enumerating alternative plans for evaluating the expression; typically, an optimizer considers a subset of all possible plans because the number of possible plans is very large.
b) Estimating the cost of each enumerated plan, and choosing the plan with the least estimated cost.

We can study more about select, project operations in relational algebra in Unit 7.

Self-Assessment Questions – 4

18. The goal of a query optimizer is to find a good evaluation plan for a given__________.
19. Optimizing a relational algebra expression involves____________basic steps.

6. SYSTEM CATALOG IN A RELATIONAL DBMS

We can store a relation using one of several alternative file structures, and we can create one or more indexes each stored as a file on every relation. Conversely, in a relational DBMS, every file contains either the tuples in relation or the entries in an index. The collection of files corresponding to users' relations and indexes represents the *data* in the database.

A fundamental property of a database system is that it maintains a description of all the data that it contains. A relational DBMS maintains information about every relation and index that it contains. The DBMS also maintains information about views, for which no tuples are stored explicitly; rather, a definition of the view is stored and used to compute the tuples that belong in the view when the view is queried. This information is stored in a collection of relations, maintained by the system, called the **catalog relations**; an example of a catalog relation is shown in Figure 5.8. The catalog relations are also called the **system catalog**, the *catalog*, or the **data dictionary**. The system catalog is sometimes referred to as **metadata**;that is, not data, but descriptive information about the data. The information in the system catalog is used extensively for query optimization.

6.1 Information Stored in The System Catalog

Let us consider what is stored in the system catalog. At a minimum, we havesystem-wide information, such as the size of the buffer pool and the page size, and the following information about individual relations, indexes, and views:

For each relation:

- Its *relation name*, the *file name* (or some identifier), and the fi*le structure*
- (e.g., heap file) of the file in which it is stored.
- The *attribute name* and *type* of each of its attributes.
- The *index name* of each index on the relation.
- The *integrity constraints* (e.g., primary key and foreign key constraints)on the relation.

For each index:

- The *index name* and the *structure* of the index.
- The *search key* attributes.

For each view:

- Its *view name* and *definition*.

In addition, statistics about relations and indexes are stored in the system catalogs and updated periodically (*not* every time the underlying relations are modified). The following information is commonly stored:

Cardinality: The number of tuples *NTuples(R)* for each relation *R*.
Size: The number of pages *NPages(R)* for each relation *R*.
Index Cardinality: Number of distinct key values *NKeys(I)* for each index *I*.
Index Size: The number of pages *INPages(I)* for each index *I*.
Index Height: The number of nonleaf levels *IHeight(I)* for each tree index *I*.
Index Range: The minimum present key value *ILow(I)* and the maximumpresent key value *IHigh(I)* for each index *I*.

6.2 How Catalogs Are Stored

A very elegant aspect of a relational DBMS is that the system catalog is itself a collection of relations. For example, we might store information aboutthe attributes of relations in a catalog relation called Attribute Cat:
Attribute Cat(*attr name:* string, *rel name:* string, *type:* string, *position:* integer)

Suppose that the database contains two relations:
Students(*sid:* string, *name:* string, *login:* string, *age:* integer, *gpa:* real)

Faculty (*fid:* string, *fname:* string, *sal:* real)

Figure 5.8 shows the tuples in the Attribute Cat relation that describe the attributes of these two relations. Notice that in addition to the tuples describing Students and Faculty, other tuples (the first four listed) describe the four attributes of the Attribute Cat relation itself! These other tuples illustrate an important point: the catalog relations describe all the relations inthe database, *including* the catalog relations themselves. When information about a relation is needed, it is obtained from the system catalog. Of course,at the implementation level, whenever the DBMS needs to find the schema of a *catalog* relation, the code that retrieves this information must be handled specially.

attr_name	*rel_name*	*type*	*position*
attr_name	Attribute_cat	string	1
rel_name	Attribute_cat	string	2
type	Attribute_cat	string	3
position	Attribute_cat	integer	4
sid	Students	string	1
name	Students	string	2
login	Students	string	3
age	Students	integer	4
gpa	Students	real	5
fid	Faculty	string	1
fname	Faculty	string	2
sal	Faculty	real	3

Fig 5.8: An Instance of the Attribute Cat Relation

The fact that the system catalog is also a collection of relations is very useful. For example, catalog relations can be queried just like any other relation, using the query language of the DBMS! Further, all the techniques available for implementing and managing relations apply directly to catalog relations. The choice of catalog relations and their schemas is not unique and is made by the implementor of the DBMS. Real systems vary in their catalog schema design, but the catalog is always implemented as a collection of relations, and it essentially describes all the data stored in the database.

Self-Assessment Questions – 5

20. The catalog relations are also called the__________catalog.
21. In a catalog related to view, the information storages are view nameand __________.
22. Index___________means Number of distinct key values *NKeys(I)* foreach index *I*.

7. SUMMARY

In this unit, we had an informal look at the relational model initially. We see that the main construct for representing data in the relational model is a **relation** that consists of a **relation schema** and a **relation instance**. We also dealt with relational database management systems and discussed RDBMS properties. We also briefly presented about the entity-relationship model as a tool for analyzing the semantic features of an application that are independent of events. Finally, we discussed the optimization and system catalogs in relational database management systems. We dealt with the catalog relations also known as system catalog, the *catalog*, or the data dictionary. The system catalog is sometimes referred to as metadata which is not data, but descriptive information about the data.

8. TERMINAL QUESTIONS

1. Create a relational schema to hold information about insurance policies and policyholders. Insurance policies have properties policyNo, startDate, premium, renewalDate, policyType. Policyolders are characterized by holderNo, holderName, holderAddress and holderTelno.
    a. What would be suitable primary keys in this database?
    b. How do we relate insurance policy information with policyholderinformation?
    c. Declare a suitable domain for policy type.
2. What is the goal of query optimization? Why is it important?
3. What information is stored in the system catalogs?
4. What are the benefits of making the system catalogs relations?

9. ANSWERS

Self Assessment Questions

1. relation schema
2. records
3. domain constraints
4. Structures
5. Manipulate
6. Consistency
7. Table
8. Primary
9. Field
10. SQL
11. data storage
12. join
13. sharing
14. redundancy
15. Multivalued
16. key
17. unique
18. query
19. two
20. System
21. Definition
22. Cardinality

Terminal Questions

1. (a) You can create a relational schema of your own. Although in thiscase, Policy No. would be a suitable primary key.
   (b) Insurance policy information and policy holder information can berelated by Fields and Tuples.
   (c) You can declare it in your own or give the domain name of existingpolicy type of any insurance company. (Refer entire unit for details)

2. The goal of a query optimizer is to find a good evaluation plan for agiven query. (Refer section 5)

3. Information for each relation (like *relation name*, the *file name,* fi*le structure, t*he *attribute name,* and *type* of each of its attributes, the *index name* of each index on the relation, the *integrity constraints* on the relation) are stored in the system catalogs. (Refer section 6.1 for detail)

4. The main benefits of making the system catalogs relations is that the catalog relations can be queried just like any other relation, using the query language of the DBMS. (Refer section 6.1 for detail)