RDBMS and SQL

By Rahul Barve

Objectives

- Introduction to RDBMS, Its Need
- Data Normalization
- Introduction to SQL, Types of SQL
- Working with Tables, Fetching Records
- Using Operators and Predicates
- SQL Functions
- Understanding Constraints
- Clauses and Joins
- DB Objects

Introduction to RDBMS

By Rahul Barve

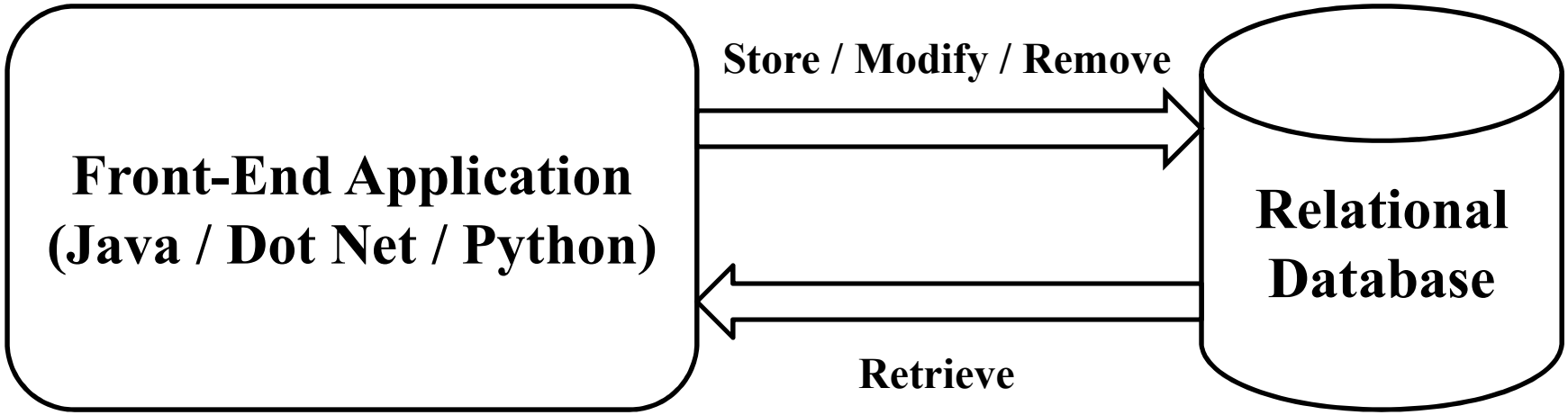
Introduction to RDBMS

- A Relational Database Management System is a special system software that is used to manage the organization, storage, access, security and integrity of a data.

RDBMS

- Allows application systems to emphasize upon the user interface, data validation and screen navigation.
- Whenever there is a need to add, modify, delete or display data, the application system simply makes a "call" to the RDBMS.

RDBMS


Front-End Application
(Java / Dot Net / Python)
Store / Modify / Remove
Retrieve
Relational
Database

Why RDBMS

By Rahul Barve

Why RDBMS

- Since a data is simply stored in a tabular format, retrieval of the data becomes easy.
- Relational model helps in reducing the redundancy.
- It makes possible to apply validation rules on the data with the help of constraints.
- It makes possible to acquire enterprise level services for data management.

Relational Database Services

By Rahul Barve

Relational Database Services

- Simple Design
- Relationships
- Constraints
- Security
- Efficient Searching and Sorting
- Transaction Isolation
- Concurrency
- Locking

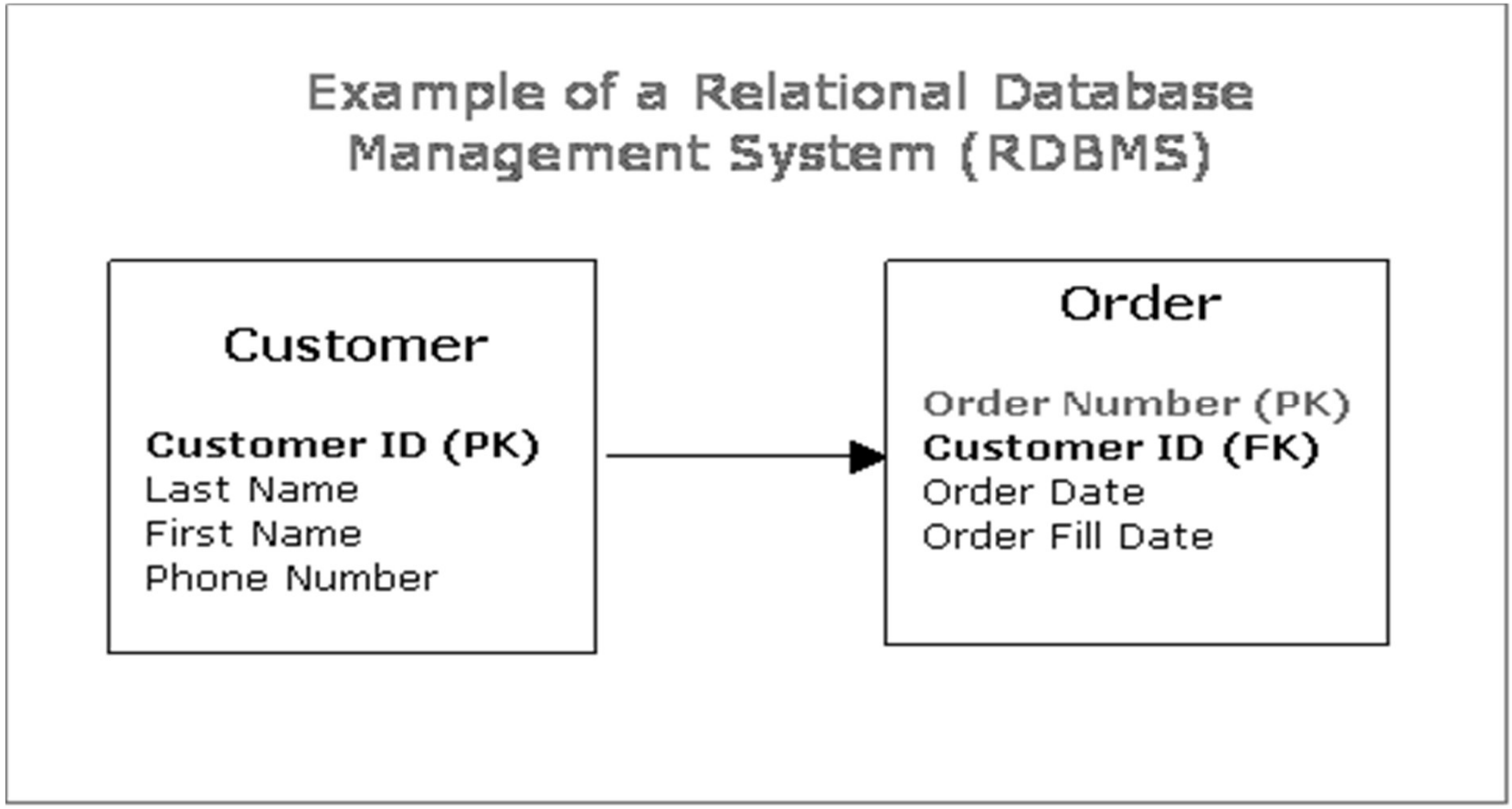
Relational Database Servers

- Oracle by Oracle Corporation
- SQL Server by Microsoft
- DB2 by IBM
- MySQL by Oracle Coporation

Database Design

- A relational database stores an information in a set of `tables`, each of which contains a unique identifier known as a `primary key`.
- These tables are further related to one another by using `foreign keys`.

Relational Database Design



Data Normalization

By Rahul Barve

Data Normalization

- Data normalization is a technique of organizing the data using a systematic approach of decomposing tables to reduce data redundancy.
- Normalization usually involves dividing large tables into smaller ones and defining relationships among them.

Data Normalization

- Data normalization is divided into 3 categories:
  - 1st Normal Form (1NF)
  - 2nd Normal Form (2NF)
  - 3rd Normal Form (3NF)

Data Normalization

- Table without normalization:

Roll No	Name	Branch	HOD	Phone No
1	Bruce	CS	Thomas	856-433-8317
2	Harry	CS	Thomas	856-433-8317
3	Maria	CS	Thomas	856-433-8317
4	Nuria	CS	Thomas	856-433-8317
5	Andrew	CS	Thomas	856-433-8317

Data Normalization – 1NF

By Rahul Barve

Data Normalization – 1NF

- It is the minimum requirement of a Database design otherwise it is considered to be a poor database design.
- Basic Rule: A column must contain a single value.

Data Normalization – 1NF

- Poor Database Design

Roll No	Name	Subject
1	Bruce	Java
2	Harry	Java, Angular
3	Maria	Angular, React
4	Nuria	SQL
5	Andrew	Python

Data Normalization – 1NF

- Database Design with 1NF

Roll No	Name	Subject
1	Bruce	Java
2	Harry	Java
2	Harry	Angular
3	Maria	Angular
3	Maria	React
4	Nuria	SQL
5	Andrew	Python

Data Normalization – 2NF

By Rahul Barve

Data Normalization – 2NF

- There are 2 conditions need to be satisfied so that the tables can be said to be in the 2nd normal form:
  - The tables must be in the 1st normal form.
  - There should not be any partial dependency of any column on a primary key.

Data Normalization – 2NF

- Student_Master

Student_ID	Name	Country
1	Bruce	USA
2	Harry	England
3	Nuria	Spain

Data Normalization – 2NF

- Course_Master

Course_ID	Name
1	Core Java
2	Java EE
3	Angular

Data Normalization – 2NF

- Score_Details

Primary Key

Score_ID	Student_ID	Course_ID	Score	Cost_$
1	1	1	87	400
2	2	3	75	550
3	1	2	77	475
4	3	3	82	550
5	2	2	80	475

Data Normalization – 2NF

- In the `Score_Details` table, column `Cost` indicates a partial dependency.
- Ideally the `Cost` column has to be a part of `Course_Master` table.

Data Normalization – 3NF

By Rahul Barve

Data Normalization – 3NF

- There are 2 conditions need to be satisfied so that the tables can be said to be in the 3rd normal form:
  - The tables must be in the 2nd normal form.
  - There should not be transitive dependency.

Data Normalization – 3NF

- Score_Details

Score_ ID	Student_ ID	Course_ ID	Score	Exam	Total Marks

Data Normalization – 3NF

- In the `Score_Details` table, column `Total_Marks` depends upon the type of the exam such as Theory or Practical i.e. `Exam` column which is a non-prime attribute.

Data Normalization – 3NF

- Ideally columns `Exam` and `Total_Marks` must be taken away from `Score_Details` and maintained in a separate table e.g. `Exam_Details`.

SQL

By Rahul Barve

SQL

- SQL stands for Structured Query Language.
- A query language used for storing and managing data in RDBMS.

SQL

- SQL commands are divided into 5 categories:
  - DDL
  - DQL
  - DML
  - DCL
  - TCL

DDL

- Data Definition Language.
  - All DDL commands are auto-committed.
  - Responsible for creating, removing or altering database objects.
  - `CREATE, ALTER, DROP, TRUNCATE`

DQL

- Data Query Language.
  - Used to retrieve data from the database tables.
  - Uses query options and conditions for fine tuning the results.
  - `SELECT …… FROM…..`

DML

- Data Manipulation Language.
  - DML commands are by default not auto-committed.
  - Used to perform manipulation on the existing data.
  - `INSERT, UPDATE and DELETE`

DCL

- Data Control Language.
  - Used to give permissions for data access with privileges.
  - `GRANT and REVOKE`

TCL

- Transaction Control Language.
  - Used to control the transactions using the commands `COMMIT` and `ROLLBACK`.

Retrieving Records

By Rahul Barve

Retrieving Records

- To retrieve the records from database table, `SELECT…..FROM` query is used.
- Syntax:
  - Selecting all columns

```
select * from <table-name>;
```

  - Selecting specific columns

```
select column1, column2, … from
<table-name>;
```

Retrieving Records

- It's also possible to use alias while retrieving records.
- It can be used especially when data is to be fetched from 2 or more tables.
- Syntax:

```
select
<alias>.<col1>,<alias>.<col2>,
… from <table-name> <alias>;
```

Operators

Operators

- An operator is a reserved word or a character used primarily in a `WHERE` clause to perform arithmetic operations and comparisons.
- Operators are divided into 2 types:
  - Arithmetic

    `+, -, *, /`
  - Comparison

    `=, !=, <, >, <=, >=, <>`

Concatenation Operator

- Used to join the values of the columns.
- Syntax:

```
select <col1>||'     '||<col2>
from <table-name> <alias>;
```

DISTINCT

- Used to retrieve only unique values from the database column.
- Syntax:

```
select DISTINCT <column-name> from
<table-name>
```

Displaying Table Structure

- To display a table structure, `DESCRIBE` command is used.
- E.g.

```
DESCRIBE EMP
DESC EMP
```

Restricting Rows

By Rahul Barve

WHERE Clause

- `SELECT…..FROM….` query always retrieves all the records.
- To retrieve specific records based on the given criterion, `WHERE` clause is used.
- Syntax:

```
select……from <table-name>
where <condition>;
```

Predicates

By Rahul Barve

Predicates

- SQL Predicates are found on the tail end of clauses, functions and SQL expression inside the existing query statements.

Predicates

- LIKE
- AND
- OR
- IN
- BETWEEN
- IS NULL
- NOT

LIKE

- Used to compare values of a column especially of type `varchar2` against some pattern specified using wildcard characters.
- Wildcard Characters
  - % - To match zero or more characters.
  - _ - To match a single character.

AND

- Used to combine multiple conditions specified in the `WHERE` clause and evaluates to boolean TRUE if all conditions are satisfied.

OR

- Used to combine multiple conditions specified in the `WHERE` clause and evaluates to boolean TRUE if any one of the conditions is satisfied.

IN

- Used to compare a value of the column for equality to a list of literal values that have been specified.
- Syntax:

```
select ….from…. where
 <column-name> in
(value1, value2, …);
```

BETWEEN

- Used to check whether a value of a column exists within a given range or not.
- Syntax:

```
select ….from…. where
 <column-name> between
<minvalue> AND <maxvalue>
```

IS NULL

- Used to compare a value with NULL .

NOT

- A predicate used for negation.
- It can be used in conjunction with other predicates e.g. `LIKE`, `IN`, `BETWEEN`, `EXISTS`, `IS NULL` etc.