ITE1003	**Database Management Systems**	**L**	**T**	**P**	**J**	**C**
		2	**0**	**2**	**4**	**4**

Pre-requisite	**CSE1001**	**Syllabus version**
		1.0

Course Objectives:

- To understand the role of data, files and databases in information systems
- To impart the knowledge of data modeling techniques
- To provide the fundamentals of front-end and back-end of databases

Expected Course Outcome:

1) Explain the basic concepts of different data models, design models, query language
2) Design entity relationship diagrams to represent simple database application scenarios.
3) Convert high-level conceptual model to relational data model; populate database; formulate relational operations
4) Analyze and improve a database design by normalization
5) Apply transaction processing to speed up the query execution and make proper transaction in a multiuser environment.
6) Understand the Security and recovery measures in the database
7) Apply Query processing techniques to optimize the performance.
8) Populate and query a database using SQL DML/DDL commands in an application.

Student Learning Outcomes (SLO):	**2, 5, 6**
[2]	Having a clear understanding of the subject related concepts and of contemporary issues
[5]	Having design thinking capability
[6]	Having an ability to design a component or a product applying all the relevant standards and with realistic constraints

Module:1	**Fundamental Concepts and Architecture**	**3 hours**

Introduction to database system, Characteristics of the Database Approach, Actors on the Scene, Workers behind the Scene, Advantages of using the DBMS Approach, Data Models, Schemas, and Instances, Three-Schema Architecture and Data Independence, The Database System Environment, Centralized and Client/Server Architectures for DBMSs, Classification of Database Management Systems

Module:2	**Conceptual Database Design**	**4 hours**

High-Level Conceptual Data Models for Database Design, Entity Types, Entity Sets, Attributes, and Keys, Relationship Types, Relationship Sets, Roles, and Structural Constraints, Weak Entity Types, ER Diagrams, Naming Conventions, and Design Issues, Relationship Types of Degree Higher than Two, EER diagrams

<table>
<tr><td colspan="4"></td></tr>
<tr><td><b>Module:3</b></td><td><b>Relational Database Design</b></td><td colspan="2"><b>5 hours</b></td></tr>
<tr><td colspan="4">Relational Model Constraints, Update Operations, Dealing with Constraint Violations, Relational Algebra, Unary Relational Operations: Operations from Set Theory, Binary Relational Operations, Additional Relational Operations, Database Design Using ER-to-Relational Mapping</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td><b>Module:4</b></td><td><b>Normalization Theory</b></td><td colspan="2"><b>4 hours</b></td></tr>
<tr><td colspan="4">Informal Design Guidelines for Relation Schemas, Functional Dependencies, Inference Rules, Equivalence, and Minimal Cover, Properties of Relational Decompositions, Algorithms for Relational Database Schema Design, Normal Forms Based on Primary Keys, Boyce-Codd Normal Form</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td><b>Module:5</b></td><td><b>Transaction and Concurrency</b></td><td colspan="2"><b>4 hours</b></td></tr>
<tr><td colspan="4">Introduction to Transaction Processing, Desirable Properties of Transactions, Characterizing Schedules Based on Serializability, Concurrency, Two-Phase Locking Techniques for Concurrency Control, Concurrency Control Based on Timestamp Ordering, Multiversion Concurrency Control Techniques</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td><b>Module:6</b></td><td><b>Recovery and Security</b></td><td colspan="2"><b>4 hours</b></td></tr>
<tr><td colspan="4">Recovery Concepts, NO-UNDO/REDO Recovery Based on Deferred Update, Recovery Techniques Based on Immediate Update, Shadow Paging, ARIES Recovery Algorithm, Security issues- Discretionary, Mandatory</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td><b>Module:7</b></td><td><b>Query Processing and Indexing</b></td><td colspan="2"><b>4 hours</b></td></tr>
<tr><td colspan="4">Query Execution plan, Basic algorithms for query execution, Heuristic Query Optimization technique, sparse and dense index, primary, secondary and clustered index, B Tree Vs. Hash Index</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td><b>Module:8</b></td><td><b>Contemporary issues:</b></td><td colspan="2"><b>2 hours</b></td></tr>
<tr><td colspan="4"></td></tr>
<tr><td></td><td><b>Total Lecture hours:</b></td><td colspan="2"><b>30 hours</b></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td colspan="4"><b>Text Book(s)</b></td></tr>
<tr><td>1.</td><td colspan="3">Ramez Elmasri and Shamkant B.Navathe, Fundamentals of Database Systems, Pearson Education,7th edition, 2013</td></tr>
<tr><td colspan="4"><b>Reference Books</b></td></tr>
<tr><td>1.</td><td colspan="3">Raghu Rama Krishnan, Database Management Systems, Tata Mcgraw Hill, 6th edition, 2010.</td></tr>
<tr><td>2.</td><td colspan="3">Abraham Silberschatz, Henry F.Korth and S.Sudarshan, Database System Concepts, Tata Mc Graw Hill, 6th edition, 2011.</td></tr>
<tr><td>3.</td><td colspan="3">Carlos Coronel and Steven Morris, Database System Design and Implementation, cennage learning, 11th edition, 2013.</td></tr>
<tr><td>4.</td><td colspan="3">Bob Bryla and Kevin Loney, Oracle Database 12c The complete Reference, Tata McGraw Hill, 1st edition, 2013.</td></tr>
<tr><td colspan="2"><b>List of Challenging Experiments (Indicative)</b></td><td colspan="2"></td></tr>
</table>

Railway Reservation System -(Redesigning IRCTC database) **Train**(train Number, name, source, destination,start_time, reach_time, traveltime, distance, class, days, type) **Ticket**(PNRNo, Transactionid, from_station, To_station, date_of_journey, class date_of_booking, total_ticket_fare, train number) **Passenger**(PNR No, Serial no, Name, Age, Reservation_status) **Train_Route**(Train_No, route_no, station_code, name, arrival_time, depart_time, distance, day) **Train_Ticket_fare**(Train_No, class, base_fare, reservation_charge, superfast_charge, other_charge, tatkal_charge, service_tax)	
1.	Create all the tables specified above. Make underlined columns as primary key.(use number, number(m,n), varchar(n), date, time, timestamp data types appropriately) Insert atleast 5 rows to each table. (Check www.irctc.co.in website for actual data) 1. Use Interactive insertion for inserting rows to the table. 2. Use ADT (varray) for class and days column in Train table.
2.	Write simple DDL/DML Queries to 1. Remove all the rows from Passenger table permanently. 2. Change the name of the Passenger table to Passenger_Details. 3. List all train details. 4. List all passenger details. 5. Give a list of trains in ascending order of number. 6. List the senior citizen passengers details. 7. List the station names where code starts with 'M'. 8. List the trains details within a range of numbers. 9. Change the super fast charge value in train fare as zero, if it is null. 10. List the passenger names whose tickets are not confirmed. 11. List the base_fare of all AC coaches available in each train. Find the ticket details where transaction id is not known. 1) Use Interactive updation for updating the seat no for particular PNR NO. 2) Find the train names that are from Chennai to Mumbai, but do not have the source or destination in its name. 3) Find the train details that are on Thursday (Use the ADT column created).
3.	Create (Alter table to add constraint) the necessary foreign keys by identifying the relationships in the table. 1) Add a suitable constraint to train table to always have train no in the range 10001 to 99999. 2) Add a suitable constraint for the column of station name, so that does not take duplicates. 3) Change the data type of arrival time, depart time (date -> timestamp or timestamp to date), and do the necessary process for updating the table with new values. 4) Add a suitable constraint for the class column that it should take values only as 1A, 2A, 3A, SL, C. 5) Add a not null constraint for the column distance in train_route.
4.	Use SQL PLUS functions to. 1. Find the passengers whose date of journey is one month from today. 2. Print the train names in upper case.

	3. Print the passenger names with left padding character. 4. Print the station codes replacing K with M. 5. Translate all the LC in class column (Train_fare) to POT and display. 6. Display the fare details of all trains, if any value is ZERO, print as NULL value. 7. Display the pnrno and transaction id, if transaction id is null, print 'not generated'. 8. Print the date_of_jounrney in the format '27th November 2010'. 9. Find the maximum fare (total fare). 10. Find the average age of passengers in one ticket. 11. Find the maximum length of station name available in the database. 12. Print the fare amount of the passengers as rounded value. 13. Add the column halt time to train route. 14. Update values to it from arrival time and depart time. High Level: 15. Update values to arrival time and depart time using conversion functions. 16. Display the arrival time, depart time in the format HH:MI (24 hours and minutes).
5.	Write Queries to. Use SET Operators 1. Find the train numbers for which reservation have not yet been made. 2. Find the train names that donot have a first AC class coach. 3. Print all the PNR nos available in the database. 4. Find passenger names who have booked to 'Pune'. Use Nested Query(in Operators) 1. Find the train names that stop in 'Katpadi'. 2. Find the train names that are superfast and the service tax is zero. 3. Find the Passenger name who have booked for the train that starts from 'Chennai'. 4. Find the trains names that have all the AC coaches and the base fare is less than 3000 for each case. Use Join Query 1. Find the train names that stop in 'Katpadi'. 2. Find the train names that are superfast and the service tax is zero. 3. Find the Passenger name (and train name) who have booked for the train that starts from 'Chennai'. 4. Display the trains names, each type of class and the total fare for each type of class. 5. Display all the train details and the ticket details (if booked any). 6. Create a sequence to provide values for the PNR no. 7. Write a query for full outer join using any of the tables above.
6.	Write Queries to. Use Coorelated (and nested) Query 1. Find the train names for which ten tickets have been reserved. 2. Find the trains that have more than ten substations. 3. Find the passengers who do not pass through 'Mettupalam'. 4. Find passengers who have booked for super fast trains. Complex queries(use groupby/groupby having/join/nested)

	1. Take the start station code and end station code and display the train details. 2. List the train names and the number of sub stations it has. 3. List the stations where all types of trains stop. 4. List the trains names that has atleast four bookings. 5. Create a table cancellation history (Insert values from ticket and passenger table). 6. Create a table for all the train numbers and class available in train_ticket_fare with total seats. 7. Find the station name that has highest number of trains stopping at.
7.	1. Write a simple PL/SQL block to. 1. Print the fibonacci series. 2. Print the factorial of a given number. 3. Print 'NOT confirmed' based on the reservation status, of a particular passenger. 4. Print the total seats available for a particular train and for a particular class. 2. Write a cursor for the following. 1. Retrieve the passenger details for “x” train number and given journey date. 2. Display the train name (once) and the substation names. 3. Display the fare details of a particular train(use basic exceptions) 4.Write a cursor to update the reservation status of the passengers(generate seat number, if seats have reached maximum, put waiting list number (30% of total seats), if waiting list number reaches maximum, put PQWL(10%of total seats), RAC-20%)
8.	1. Write a PL/SQL procedure to. 1. List the details of passengers who has reserved next to “Mr. X”. 2. PNR No. of a passengers for a given source and a destination. 2. Write a PL/SQL function to. 1. Get the PNRNo and return the total ticket fare. 2. Get the Passenger name, train no and return the total journey time in hours and minutes.
9.	Write a Trigger for the following: 1) When a passenger cancels a ticket, do the necessary process and update the cancellation history table. 2) When train number is changed, update it in referencing tables. 3) When a passenger record is inserted reservation status should be automatically updated.
10.	1) Use TCL commands for your transactions. (commit,rollback,savepoint) 2) Create a role named 'clerk', and give permisson for him to select only the trains starting from 'Katpadi' along with fare details. 3) Create a nested table containing trainno,name,source,destination and passengers who have booked for it (PNR no,sno, name,age). Find the passengers whose name start with 'S' and train starts from 'Katpadi'

		Total Laboratory Hours	30 hours
Recommended by Board of Studies	0 4-12-2015		
Approved by Academic Council	No. 39	Date	12-12-2015