Database Management System

Lab Manual

Department of Computer Science and Engineering
The NorthCap University, Gurugram

Database Management System

Lab Manual

CSL 214

Ms. Kanika Gupta

Dr. Prachi

Dr. Anuradha Dhull

Dr. Priyanka Vashisht

Department of Computer Science and Engineering

NorthCap University, Gurugram- 122001, India

Session 2020-21

Published by:

School of Engineering and Technology

Department of Computer Science & Engineering

The NorthCap University Gurugram

• Laboratory Manual is for Internal Circulation only

Copying or facilitating copying of lab work comes under cheating and is considered as use of unfair means. Students indulging in copying or facilitating copying shall be awarded zero marks for that particular experiment. Frequent cases of copying may lead to disciplinary action. Attendance in lab classes is mandatory.

Labs are open up to 7 PM upon request. Students are encouraged to make full use of labs beyond normal lab hours.

PREFACE

Database Management System Lab Manual is designed to meet the course and program requirements of NCU curriculum for B.Tech II year students of CSE branch. The concept of the lab work is to give practical experience for basic lab skills to students. It provides the space and scope for self-study so that students can come up with new and creative ideas.

The Lab manual is written on the basis of "teach yourself pattern" and expected that students who come with proper preparation should be able to perform the experiments without any difficulty. Brief introduction to each experiment with information about self-study material is provided. The laboratory exercises will include familiarization with SQL and NoSQL commands to define, manipulate, retrieve and manage data in a database. The course will enable stduents to create a database and devise queries for extracting information from the database using Relational Algebra and SQL. The students will be able to appreciate the importance of handling unstructured data using NoSQL and apply the concepts of DBMS for developing a backend for a non-trivial project.

At the start of each experiment a question bank for preparation and practice is suggested which may be used to test the basic understanding of the students about the experiment. Students are expected to come thoroughly prepared for the lab. General disciplines, safety guidelines and report writing are also discussed.

The lab manual is a part of curriculum for the TheNorthCap University, Gurugram. Teacher's copy of the experimental results and answer for the questions are available as sample guidelines.

We hope that lab manual would be useful to students of CSE, IT, ECE and BSc branches and author requests the readers to kindly forward their suggestions / constructive criticism for further improvement of the work book.

Author expresses deep gratitude to Members, Governing Body-NCU for encouragement and motivation.

Authors
The NorthCap University
Gurugram, India

CONTENTS

COURSE TEMPLATE

1. Department:	**Department of Computer Science and Engineering**		
2. Course Name: Database Management Systems	**3. Course Code**	**4. L-T-P**	**5. Credits**
	CSL214	3-0-2	4
6. Type of Course (Check one):	Programme Core ✓	Programme Elective ☐	Open Elective ☐
7. Pre-requisite(s), if any: None			
8. Frequency of offering (check one): Odd ☐	Even ✓	Either semester ☐	Every semester ☐

9. Brief Syllabus:

Databases form the core of all major applications – finance, social, administrative, education etc. Organizations work on large volumes of data every day, introducing the need to have database management systems to easily identify, extract, store and transform details in the database. This course will explore concepts and principles of DBMS, database design, data modeling, database implementation, and database management through various assignments and projects. By the end of this course, the student will be able to work as a database engineer by designing, developing and maintaining the database for any project application.

Total Lecture, Tutorial and Practical Hours for this course (Take 15 teaching weeks per semester): 75

	Practice	
Lectures: 45 hours	**Tutorials:**10 hours	**Lab Work:** 20 hours

10. Course Outcomes (COs)

Possible usefulness of this course after its completion i.e. how this course will be practically useful to him once it is completed.

CO 1	Identifying contrast between traditional and modern Database Systems, thereby recognizing their applications.
CO 2	Developing conceptual database design for any real time project by defining the relationship, constraints etc. on entities.
CO 3	Applying appropriate design techniques to design a good database that meets the user requirement.
CO 4	Creating a database and devising queries for extracting information from the database using Relational Algebra and SQL.
CO 5	Applying the concepts of DBMS for developing a backend for a non-trivial project using NoSQL.
CO 6	Ability to improvise data fetching time by applying indexing concepts.
CO 7	Understanding the concepts of end-to-end transaction processing in a database.

11. UNIT WISE DETAILS **No. of Units: 7**

Unit Number: 1	Title: Introduction to Database Systems	No. of hours: 4
Content Summary: Overview of Database Management Systems, Advantages of DBMS over File Processing Systems, DBMS Vs. RDBMS, DBA roles and responsibilities, Data Independence, Architecture of Database(3-Schema Architecture, Complete architecture), Database Query Languages (DDL, DML, DCL), Relational Model Concepts: Primary Key, Unique key, Foreign key, Super Key, Alternate key, Candidate key, Constraints used in Relational Data Model including integrity constraints.		
Unit Number: 2	**Title: Conceptual Database Design**	**No. of hours: 8**
Content Summary: Data Modeling Using the Entity Relationship (ER) Model, The Enhanced Entity-Relationship (EER) Model: Entity Set, attributes and their types, Relationship Constraints (including Participation constraints and cardinality ratio), ER Diagrams, constraints and design issues, Reduction of ER and EER diagram to relational schemas.		
Unit Number: 3	**Title: Relational Database Design**	**No. of hours: 8**
Content Summary: Relational database design, Functional dependencies: Fully functional dependency, partial FD, trivial, non-trivial FD, inference rules, canonical cover, lossless join, dependency preservation, multi value dependency, Normal Forms: 1NF, 2NF, 3NF, BCNF, 4NF, 5NF, Normalization and denormalization process		
Unit Number: 4	**Title: Query Languages**	**No. of hours: 8**
Content Summary: Relational Algebra: relational operators (projection, selection , union, intersection, set difference, division, rename, Cartesian product, generalized relational algebra operators, cross product, join operators : inner vs. outer join, theta join, outer join, natural join, equijoin, self-join, complete set of relational algebra operations. SQL- Queries, Constraints, Form of SQL query, UNION, INTERSECT and EXCEPT, Nested queries, Aggregate Operators, Null values, Complex Integrity constraints in SQL and triggers.		
Unit Number: 5	**Title: Introduction to NoSQL (MongoDB)**	**No. of hours: 4**
Content Summary: Introduction to MongoDB, Datatypes, Document Data Model-Creating, Inserting, Updating and Deleting Documents, MongoDB Query Language, Sorting, Join Operations.		
Unit Number: 6	**Title: File Organization& Indexing**	**No. of hours: 6**
Content Summary: Disk Storage, Basic File Structures and Hashing: Unordered, ordered and hashed files of records, Single and multilevel indexes: primary index, secondary index, clustered, multilevel and dynamic multilevel indexes (B-Tree and B+ Tree).		
Unit Number: 7	**Title: Transaction Management & Concurrency Control**	**No. of hours: 7**
Content Summary:		

Introduction to transaction processing, ACID Properties, Concurrency control mechanisms: serializability, two phase locking protocol, basic concept of deadlock, deadlock handling, timestamp-based protocols, precedence graph to ensure serializability, different protocols in concurrency control. Database back-up and Recovery.

12. Brief Description of Self-learning components by students (through books/resource material etc.):

- Aggregation and Pagination in MongoDB

13. Books Recommended:

Textbooks:

1. Elmasri R. and Navathe S.B., Fundamentals of Database Management Systems. 6th ed. Pearson, 2010.
2. Silberschatz A., Korth H.F. and Sudarshan S., Database System Concepts. 6th ed. Mc.Graw Hill, 2010.
3. Chodorow K., MongoDB: The Definitive Guide. 2nd ed. O'Reilly Media, 2013.

Reference Books:

1. Ramakrishnan R. and Gehrke J., Database Management Systems. 3rd ed. McGraw-Hill Education, 2003.
2. Suehring S., My SQL Bible. Wiley Publishing, 2002.

Reference Websites: (nptel, swayam, coursera, edx, udemy, lms, official documentation weblink)

- https://nptel.ac.in/courses/106105175/2
- https://docs.mongodb.com/

Practice (Tutorial/Case Studies/ Industry Visit/Field Work) Content

Sr. No.	Topic	Unit Covered
1	Relational Keys	1
2	Normal Forms	3
3	Normalization	3
4	Relational Algebra	4
5	File Organization	6
6	Indexing	6
7	Serializability	7

Practical Content

Sr. No.	Title of the Experiment	Software/ Hardware Based	Unit Covered	Time Required
1	Design an ER diagram for the COMPANY database for the following set of requirements.	erdplus.com	2	2 hours
2	Design a Relational Database Design for the COMPANY database from the ER/EER diagram for the following set of requirements.		2	1 hour
3	To apply SQL integrity constraints as per the DDL statements given below for COMPANY database.	MySQL	4	2 hour
4	To familiarize with SELECT-FROM-WHERE SQL simple queries on the COMPANY database.	MySQL	4	2 hours
5	To familiarize with different JOIN operations in SQL on the COMPANY database.	MySQL	4	2 hours

6	To understand Aggregate functions and Group by Clause using SQL queries on the COMPANY database.	MySQL	4	2 hours
7	To familiarize with nested SQL queries on the COMPANY database.	MySQL	4	2 hours
8	Identifying contrast between Relational Databases and NoSQL, thereby recognizing their applications.	mongodb.com	5	1 hour
9	Create COMPANY database using NoSQL database - MongoDB.	MongodbShell	5	2 hours
10	Retrieve data from NoSQL database - MongoDB.	Mongodb Shell	5	2 hours
Value Added Experiments				
1	Sorting and Indexing of Data in COMPANY Database	Mongodb Shell	5	2 hours

Project (To be done as individual/in group): No

1. INTRODUCTION

That 'learning is a continuous process' cannot be over emphasized. The theoretical knowledge gained during lecture sessions need to be strengthened through practical experimentation. Thus, practical makes an integral part of a learning process.

The purpose of conducting experiments can be stated as follows:

- To familiarize the students with the basic concepts, programming skill development and the take home laboratory assignments mainly implementation-oriented which have to be coded in high level language. The lab sessions will be based on exploring the concepts discussed in class.
- Developing conceptual database design for any real time project by defining the relationship, constraints etc. on entities.
- Compare and contrast application of SQL and NoSQL for managing a database.
- Hands on experience on the experimental setup and software.

2. LAB REQUIREMENTS

Requirements	Details
Software Requirements	MySQL, MongoDB
Operating System	Any
Hardware Requirements	Windows and Linux: Intel 64/32 or AMD Athlon 64/32, or AMD Opteron processor 2 GB RAM 80 GB hard disk space
Required Bandwidth	NA

3. GENERAL INSTRUCTIONS

3.1 General discipline in the lab

- Students must turn up in time and contact concerned faculty for the experiment they are supposed to perform.
- Students will not be allowed to enter late in the lab.
- Students will not leave the class till the period is over.
- Students should come prepared for their experiment.
- Experimental results should be entered in the lab report format and certified/signed by concerned faculty/ lab Instructor.
- Students must get the connection of the hardware setup verified before switching on the power supply.
- Students should maintain silence while performing the experiments. If any necessity arises for discussion amongst them, they should discuss with a very low pitch without disturbing the adjacent groups.
- Violating the above code of conduct may attract disciplinary action.
- Damaging lab equipment or removing any component from the lab may invite penalties and strict disciplinary action.

3.2 Attendance

- Attendance in the lab class is compulsory.
- Students should not attend a different lab group/section other than the one assigned at the beginning of the session.
- On account of illness or some family problems, if a student misses his/her lab classes, he/she may be assigned a different group to make up the losses in consultation with the concerned faculty / lab instructor. Or he/she may work in the lab during spare/extra hours to complete the experiment. No attendance will be granted for such case.

3.3 Preparation and Performance

- Students should come to the lab thoroughly prepared on the experiments they are assigned to perform on that day. Brief introduction to each experiment with information about self-study reference is provided on LMS.
- Students must bring the lab report during each practical class with written records of the last experiments performed complete in all respect.
- Each student is required to write a complete report of the experiment he has performed and bring to lab class for evaluation in the next working lab. Sufficient space in work book is provided for independent writing of theory, observation, calculation and conclusion.
- Students should follow the Zero tolerance policy for copying / plagiarism. Zero marks will be awarded if found copied. If caught further, it will lead to disciplinary action.
- Refer **Annexure 1** for Lab Report Format.

4. LIST OF EXPERIMENTS

Sr. No.	Title of the Experiment	Software/ Hardware Based	Unit Covered	CO Covered	Time Required
1	Design an ER diagram for the COMPANY database for the following set of requirements.	erdplus.com	2	CO2	2 hours
2	Design a Relational Database Design for the COMPANY database from the ER/EER diagram for the following set of requirements.		2	CO2	1 hour
3	To apply SQL integrity constraints as per the DDL statements given below for COMPANY database.	MySQL	4	CO4	2 hour
4	To familiarize with SELECT-FROM-WHERE SQL simple queries on the COMPANY database.	MySQL	4	CO4	2 hours
5	To familiarize with different JOIN operations in SQL on the COMPANY database.	MySQL	4	CO4	2 hours
6	To understand Aggregate functions and Group by Clause using SQL queries on the COMPANY database.	MySQL	4	CO4	2 hours
7	To familiarize with nested SQL queries on the COMPANY database.	MySQL	4	CO4	2 hours
8	Identifying contrast between Relational Databases and NoSQL, thereby recognizing their applications.	mongodb.com	5	CO5	1 hour
9	Create COMPANY database using NoSQL database - MongoDB.	Mongodb Shell	5	CO5	2 hours
10	Retrieve data from NoSQL database - MongoDB.	Mongodb Shell	5	CO5	2 hours
VALUE ADDED EXPERIMENT					
1	Sorting and Indexing of Data in COMPANY Database	Mongodb Shell	5	CO5	2 hours

5. LIST OF FLIP EXPERIMENTS

1. Implementation of Aggregation Functions in MongoDB
2. Implementation of Pagination Functions in MongoDB

6. RUBRICS

Marks Distribution	
Continuous Evaluation (50 Marks)	**Project Evaluations (20 Marks)**
Each experiment shall be evaluated for 10 marks and at the end of the semester proportional marks shall be awarded out of 50.	End semester practical evaluation including Mini project (if any) carries 20 marks.
Following is the breakup of 10 marks for each **4 Marks**: Observation & conduct of experiment. Teacher may ask questions about experiment. **3 Marks:** For report writing **3 Marks:** For the 15 minutes quiz to be conducted in every lab.	

Annexure 1

Database Management System

(CSL 214)

Lab Practical Report

Faculty name:

Student name:

Roll No.:

Semester:

Group:

Department of Computer Science and Engineering

NorthCap University, Gurugram- 122001, India

Session 2020-21

INDEX

EXPERIMENT NO. 1

Student Name and Roll Number: Chirag Sardana and 19CSU071
Semester /Section: 4th/ FS A1
Link to Code: https://github.com/chiragsardana/Question-In-Java/tree/master/DBMS/Experiment%201
Date:
Faculty Signature:
Marks:

Objective(s):

Design an ER diagram for the COMPANY database for the following set of requirements.

Outcome:

The students will be able to draw conceptual database design using ERD Plus.

Problem Statement:

The COMPANY database keeps track of a company's employees, departments, and projects. Suppose that after the requirements collection and analysis phase, the database designers provide the following description of the miniworld—the part of the company that will be represented in the database.

- A department controls a number of projects, each of which has a unique name, a unique number, and a single location.
- The company is organized into departments. Each department has a unique name, a unique number, and a particular employee who manages the department. We keep track of the start date when that employee began managing the department. A department may have several locations.
- We store each employee's name, Social Security number, address, salary, sex (gender), and birth date. An employee is assigned to one department, but may work on several projects, which are not necessarily controlled by the same department. We keep track of the current number of hours per week that an employee works on each project. We also keep track of the direct supervisor of each employee (who is another employee).

- We want to keep track of the dependents of each employee for insurance purposes. We keep each dependent's first name, sex, birth date, and relationship to the employee.

Design the ER model by identifying the following from the above requirements:

i) Entities (Strong and Weak)
ii) Relationships
iii) Participation constraints
iv) Various types of attribute
v) Recursive relations
vi) Mapping cardinalities
vii) Binary/Ternary relationship
viii) Specialization/Generalization etc.

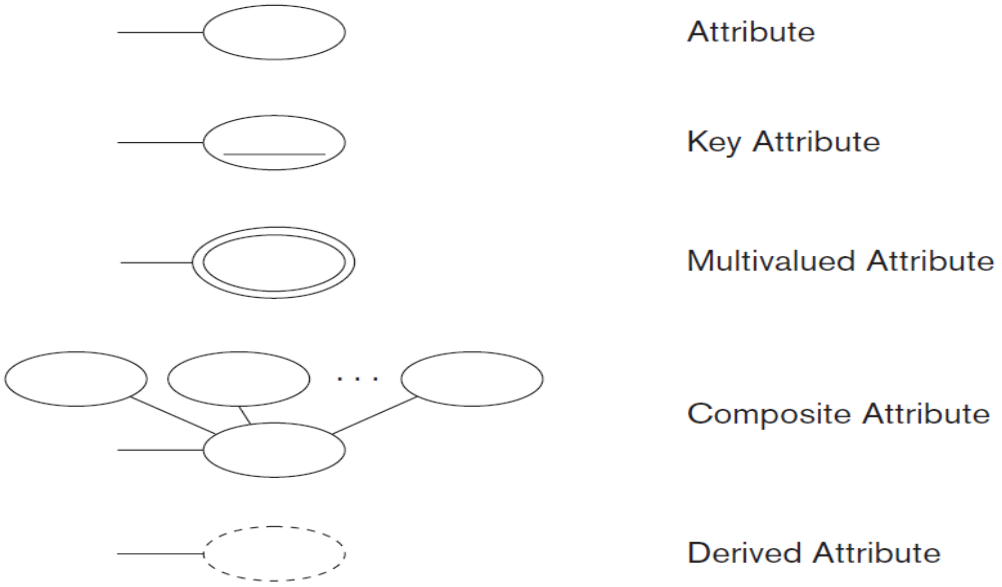
Background Study:

ER Diagram Symbols and Notations:

1) Entity:
   - Real-world object distinguishable from other objects.
   - An entity is described using a set of *attributes*.

2) Entity Set: A collection of similar entities. Eg: all employees.
   - All entities in an entity set have the same set of attributes.
   - Each entity set has a *key*.
   - Each attribute has a *domain*.

3) Attributes
   - Attributes are properties used to describe an entity.
   - Example: EMPLOYEE entity may have a Name, SSN, Address, Sex, BirthDate.



4) Relationship
   - A relationship relates two or more distinct entities with a specific meaning.
   - Relationships of the same type are grouped or typed into a relationship type.
   - 3 types of relationships : Unary, Binary & Ternary.

5) Recursive Relationship
   - A relationship with the same participating entity type in distinct roles.
   - Example: the SUPERVISION relationship

6) Structural Constraints – Semantics of Relationships
   - Cardinality Ratio : The number of instances of an entity from a relation that can be associated with the relation.
   - Participation Constraints
     - Total Participation − Each entity is involved in the relationship.
     - Partial participation − Not all entities are involved in the relationship.

7) Specialization - is the process of defining a set of subclasses of a superclass

8) Generalization
   - the reverse of the specialization process .
   - Several classes with common features are generalized into a superclass (original classes become its subclasses)

Question Bank:

Q1) Given the basic ER and relational models, which of the following is INCORRECT?

1) An attribute of an entity can have more than one value
2) An attribute of an entity can be composite
3) In a row of a relational table, an attribute can have more than one value
4) In a row of a relational table, an attribute can have exactly one value or a NULL value

Answer is : 3

The term 'entity' belongs to ER model and the term 'relational table' belongs to relational model.

1 and 2 both are true. ER model supports both multivalued and composite attributes See this for more details.

3 is false and 4 is true. In Relation model, an entry in relational table can can have exactly one value or a NULL.

Q2) Consider a directed line(-->) from the relationship set advisor to both entity sets instructor and student. This indicates _____ cardinality

1) One to many
2) One to one
3) Many to many
4) Many to one

Answer is : 2

Q3) An entity set that does not have sufficient attributes to form a primary key is termed as :

1) Strong entity set
2) Variant set
3) Weak entity set
4) Variable set

Answer is : 3

Q4) In which of the following cases an entity instance must be a member of only one subtype?

1. Disjoint with total specialisation
2. Disjoint with partial specialisation
3. Overlap with total specialisation

4. Overlap with Partial Specialisation

Answer is : 1

Q5) Which of the following indicates the maximum number of entities can be involved in a relationship?

1) Minimum cardinality
2) Maximum Cardinality
3) ERD
4) Greater Entity Count (GEC)

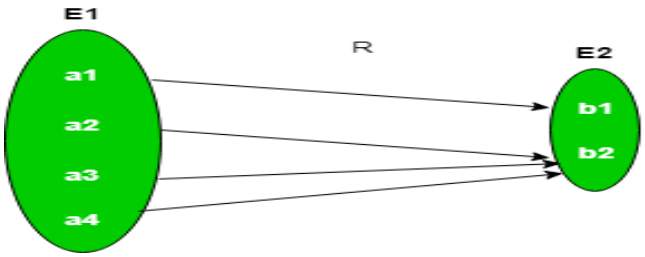
Answer is : 2

Q6) In an Entity-Relationship (ER) model, suppose R is a many-to-one relationship from entity set E1 to entity set E2. Assume that E1 and E2 participate totally in R and that the cardinality of E1 is greater that the cardinality of E2. Which one of the following is true about R?

1) Every entity in E1 is associated with exactly one entity in E2.
2) Some entity in E1 is associated with more than one entity in E2.
3) Every entity in E2 is associated with exactly one entity in E1.
4) Every entity in E2 is associated with at most one entity in E1.

Answer is: 1

Since given relation is **many to one** :



Therefore, no entity in E1 can be related to more than one entity in E2 and an entity in E2 can be related to more than one entity in E1.

Only option 1 is correct.

Q7) Which symbol denote derived attributes in ER Model?

1) Double ellipse
2) Dashed ellipse
3) Squared ellipse
4) Ellipse with attribute name underlined

Answer is : 2

Q8) How are roles specified in an ER diagram
1) By labelling the rectangles
2) By labelling the diamonds
3) Roles cannot be specified in an ER diagram
4) By labelling the lines

Answer is: 4

Q9) For a weak entity set to be meaningful, it must be associated with another entity set in combination with some of their attribute values, is called as:

1) Neighbour Set
2) Strong Entity Set
3) Owner Entity Set
4) Weak Set

Answer is: 3

Q10) Which of the following statements is FALSE about weak entity set?

1) Weak entities can be deleted automatically when their strong entity is deleted.
2) Weak entity set avoids the data duplication and consequent possible inconsistencies caused by duplicating the key of the strong entity.

3) A weak entity set has no primary keys unless attributes of the strong entity set on which it depends are included
4) Tuples in a weak entity set are not partitioned according to their relationship with tuples in a strong entity set

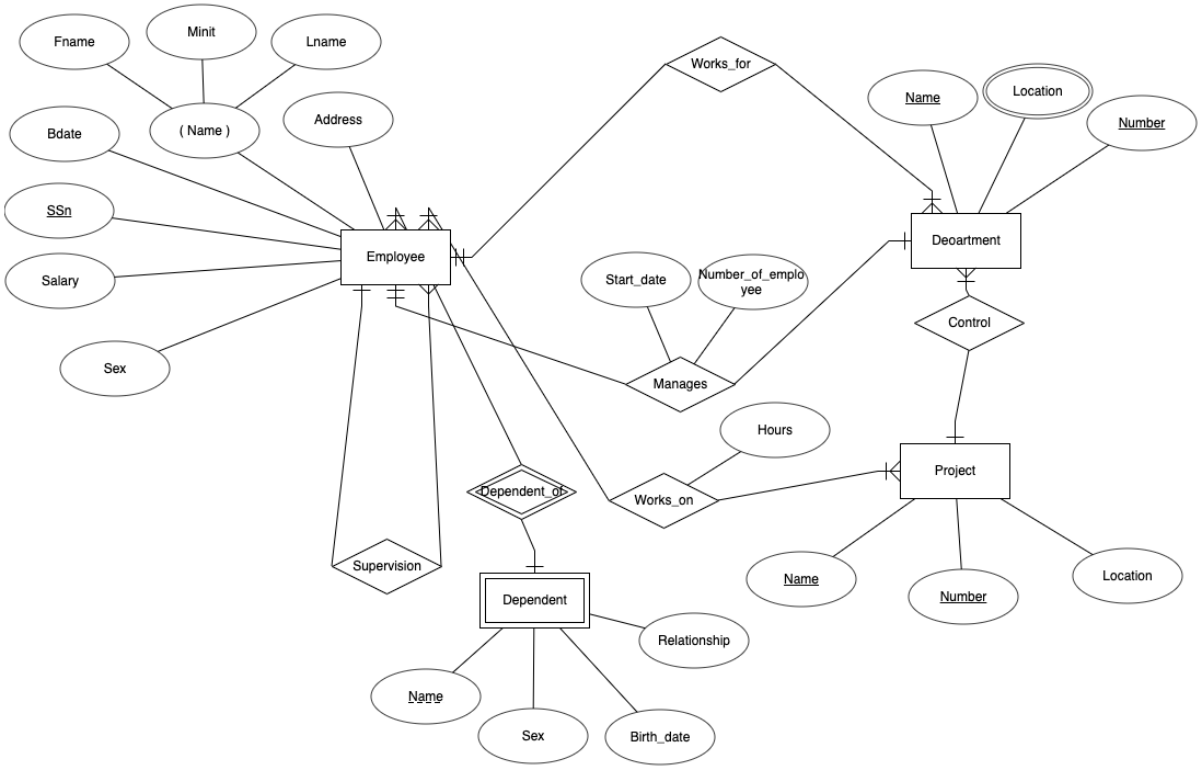
Answer is: 4

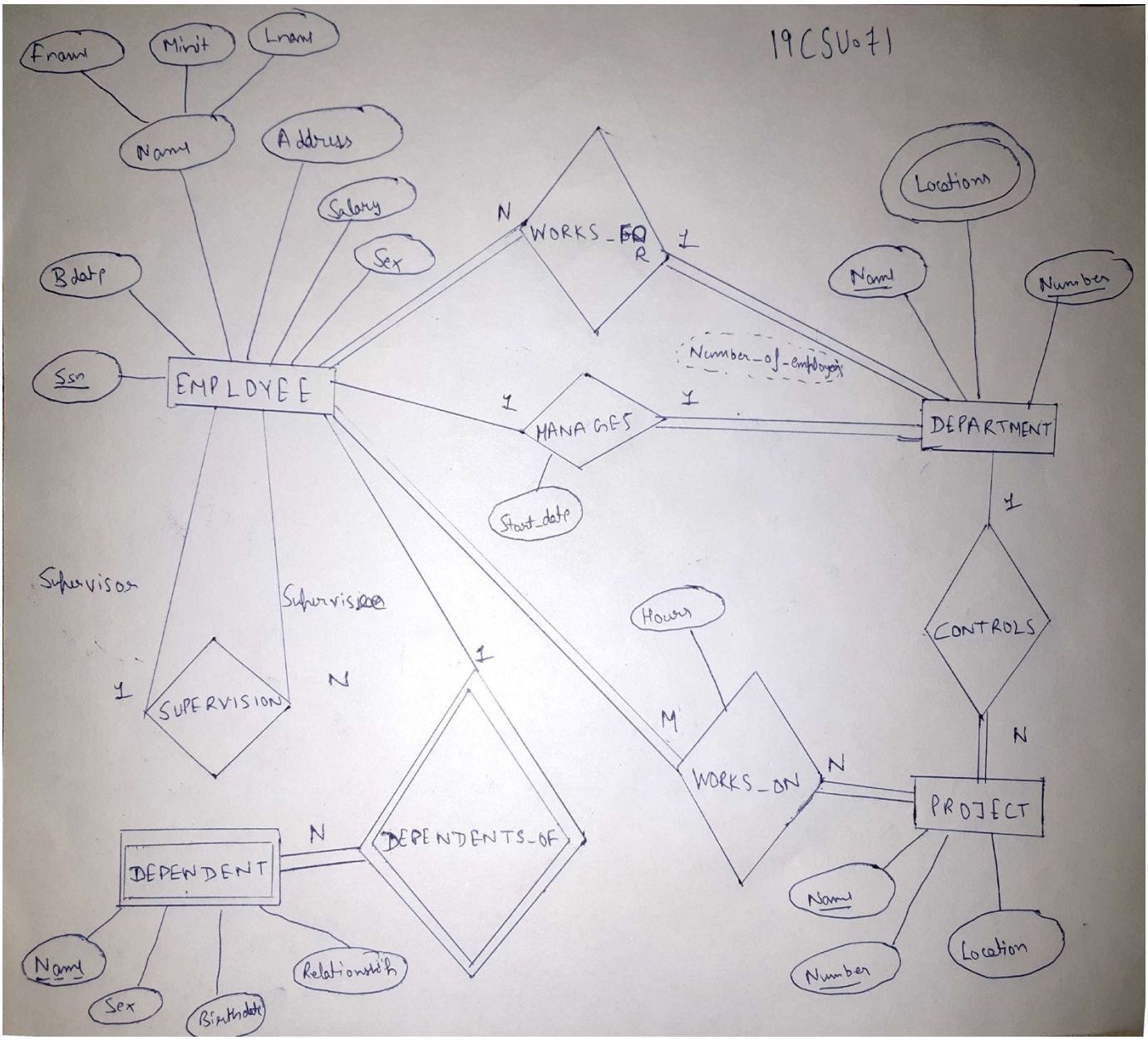
- Weak entities can be deleted automatically when their strong entity is deleted. **Correct**
- Weak entity set avoids the data duplication and consequent possible inconsistencies caused by duplicating the key of the strong entity. **Correct**
- A weak entity set has no primary keys unless attributes of the strong entity set on which it depends are included **Correct**
- Tuples in a weak entity set are not partitioned according to their relationship with tuples in a strong entity set. This is **Incorrect**, because tuples in a weak entity set are partitioned according to their relationship with tuples in a strong entity set.

Student Work Area

Algorithm/Flowchart/Code/Sample Outputs

ER/EER Diagram





Entities:

Strong: Employee, Department, Project

Weak: Dependent

Relationships:

Manages 1:1 Employee and Department

Works_for N:1 Employee and Department

Controls 1:N Department and Project

Supervision 1:N Recursive (Employee)

Works_on M:N Emloyee and Project

Dependents_of 1:N Employee and Dependent

EXPERIMENT NO. 2

Student Name and Roll Number: Chirag Sardana and 19CSU071
Semester /Section: 4th/ FS A1
Link to Code: https://github.com/chiragsardana/Question-In-Java/tree/master/DBMS/Experiment%202
Date:
Faculty Signature:
Marks:

Objective(s):

Design a Relational Database Design for the COMPANY database from the ER/EER diagram for the following set of requirements.

Outcome:

The students will be able to map the conceptual database design to logical (relational) database design.

Problem Statement:

The COMPANY database keeps track of a company's employees, departments, and projects. Map the ER/EER diagram of the COMPANY database created in the previous experiment to Relational Database Design. Follow the below mentioned steps to successfully map ER/EER to relational tables:

Step 1: Mapping of Regular Entity Types

Step 2: Mapping of Weak Entity Types

Step 3: Mapping of Binary 1:1 Relation Types

Step 4: Mapping of Binary 1:N Relationship Types

Step 5: Mapping of Binary M:N Relationship Types

Step 6: Mapping of Multivalued attributes

Step 7: Mapping of N-ary Relationship Types

Step 8: Mapping EER Model Constructs to Relations

Step 9: Options for Mapping Specialization or Generalization

Step 10: Mapping of Union Types (Categories)

Background Study:

1) Mapping of Regular Entity Types
   - For each regular (strong) entity type E in the ER schema, create a relation R that includes all the simple attributes of E. Choose one of the key attributes of E as the primary key for R. If the chosen key of E is composite, the set of simple attributes that form it will together form the primary key of R.

2) Mapping of Weak Entity Types
   - For each weak entity type W in the ER schema with owner entity type E, create a relation R and include all simple attributes (or simple components of composite attributes) of W as attributes of R.
   - Include as foreign key attributes of R the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s).

3) Mapping of Binary 1:1 Relationship Types
   - For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R. This has 3 approaches :
     - Foreign Key Approach
     - Merged Relation Option
     - Cross-reference or Relationship Relation Option

4) Mapping of Binary 1:N Relationship Types
   - For each regular binary 1:N relationship type R, identify the relation S that represent the participating entity type at the N-side of the relationship type.
   - Include as foreign key in S the primary key of the relation T that represents the other entity type participating in R.

5) Mapping of Binary M:N Relationship Types
   - For each regular binary M:N relationship type R, create a new relation S to represent R.
   - Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types; their combination will form the primary key of S.

6) Mapping of Multi-valued attributes
   - For each multi-valued attribute A, create a new relation R. This relation R will include an attribute corresponding to A, plus the primary key attribute K-as a foreign key in R-of the relation that represents the entity type of relationship type that has A as an attribute.

7) Mapping of N-ary Relationship Types
   - For each n-ary relationship type R, where n>2, create a new relationship S to represent R.
   - Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types.

8) Options for Mapping Specialization or Generalization
   - Convert each specialization with m subclasses $\{S_1, S_2,\ldots,S_m\}$ and generalized superclass C, where the attributes of C are $\{k,a_1,\ldots a_n\}$ and k is the (primary) key, into relational schemas using 1 of the following :
     1) Multiple relations-Superclass and subclasses.
     2) Multiple relations-Subclass relations only :
     3) Single relation with *one type attribute*
     4) Single relation with multiple type attributes.

9) Mapping of Union Types (Categories)
   - For mapping a category whose defining superclass have different keys, it is customary to specify a new key attribute, called a surrogate key, when creating a relation to correspond to the category.

Question Bank:

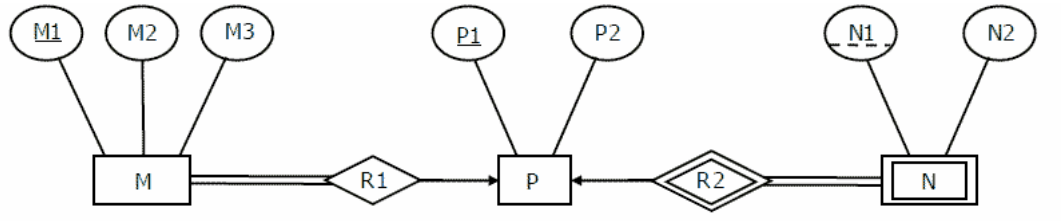
Q1) In which of the following, a separate schema is created consisting of that attribute and the primary key of the entity set.

1) A many-to-many relationship set
2) A multivalued attribute of an entity set
3) A one-to-many relationship set
4) All of the mentioned

Answer is: 2

If a multivalued dependency holds and is not implied by the corresponding functional dependency, it usually arises from this source.

Q2) Consider the following ER diagram



The minimum number of tables needed to represent M, N, P, R1, R2 is

1) 2
2) 3
3) 4
4) 5

Answer is: 2(Means Number of Table Required is 3)

M, P are strong entities hence they must be represented by separate tables.

Many-to-one and one-to-many relationship sets that are total on the many-side can be represented by adding an extra attribute to the “many” side, containing the primary key of the “one” side. (This way no extra table will be needed for Relationship sets)

M table is modified to include primary key of P side(i.e. P1). N is weak entity, and is modified to include primary key of P (i.e, P1).

Therefore there would be minimum of 3 tables with schema given below :

M (M1, M2, M3, P1)

P (P1, P2)

N (P1, N1, N2)

Q3) Consider the data given in above question. Which of the following is a correct attribute set for one of the tables for the correct answer to the above question?

1) {M1, M2, M3, P1}
2) {M1, P1, N1, N2}
3) {M1, P1, N1}
4) {M1, P1}

Answer is: 1

Q4) Let E1 and E2 be two entities in an E/R diagram with simple single-valued attributes. R1 and R2 are two relationships between E1 and E2, where R1 is one-to-many and R2 is many-to-many. R1 and R2 do not have any attributes of their own. What is the minimum number of tables required to represent this situation in the relational model?

1) 2
2) 3
3) 4
4) 5

Answer is: 2

Strong entities E1 and E2 are represented as separate tables.

In addition to that many-to-many relationships(R2) must be converted as separate table by having primary keys of E1 and E2 as foreign keys.

One-to-many relaionship (R1) must be transferred to 'many' side table(i.e. E2) by having primary key of one side(E1) as foreign key(this way we need not to make a separate table for R1).

Let relation schema be E1(a1,a2) and E2(b1,b2).

Relation E1(a1 is the key)

a1 a2

---

1 3

2 4

3 4

Relation E2(b1 is the key, a1 is the foreign key, hence R1(one-many) relationship set satisfy here)

b1 b2 a1

---

7 4 2

8 7 2

9 7 3

Relation R2 ({a1, b1} combined is the key here , representing many-many relationship R2)

a1 b1

---

1 7

1 8

2 9

3 9

Hence we will have minimum of 3 tables.

Q5) What is the min and max number of tables required to convert an ER diagram with 2 entities and 1 relationship between them with partial participation constraints of both entities?

1) Min 1 and max 2
2) Min 1 and max 3
3) Min 2 and max 3
4) Min 2 and max 2

Answer is: 3

Maximum number of tables required is 3 in case of many to many relationships between entities. Minimum number of tables is 1 in case of unary relationship and total participation of atleast one entity. But in case of partial participation of both entities, minimum number of tables required is 2.

Q6) Which of the following has each related entity set has its own schema and there is an additional schema for the relationship set.

1) A many-to-many relationship set
2) A multivalued attribute of an entity set
3) A one-to-many relationship set
4) All of the mentioned

Answer is: 1

If a multivalued dependency holds and is not implied by the corresponding functional dependency, it usually arises from this source.

Q7) For each attribute of a relation, there is a set of permitted values, called the _____ of that attribute.

1) Domain

2) Relation
3) Set
4) Schema

Answer is: 1

The values of the attribute should be present in the domain. Domain is a set of values permitted.

Q8) Let M and N be two entities in an E-R diagram with simple single value attributes. R1 and R2 are two relationship between M and N, where as R1 is one-to-many and R2 is many-to-many. The minimum number of tables required to represent M, N, R1 and R2 in the relational model are _____.

1) 4
2) 6
3) 7
4) 3

Answer is: 4

Strong entities E1 and E2 are represented as separate tables.

In addition to that many-to-many relationships(R2) must be converted as separate table by having primary keys of E1 and E2 as foreign keys.

One-to-many relaionship (R1) must be transferred to 'many' side table(i.e. E2) by having primary key of one side(E1) as foreign key(this way we need not to make a separate table for R1).

Let relation schema be E1(a1,a2) and E2(b1,b2).

Relation E1(a1 is the key)

a1 a2

---

1 3

2 4

3 4

Relation E2(b1 is the key, a1 is the foreign key, hence R1(one-many) relationship set satisfy here)

b1 b2 a1

---

7 4 2

8 7 2

9 7 3

Relation R2 ({a1, b1} combined is the key here , representing many-many relationship R2)

a1 b1

---

1 7

1 8

2 9

3 9

Hence we will have minimum of 3 tables.

Q9) Database _____ which is the logical design of the database, and the database _____ which is a snapshot of the data in the database at a given instant in time.

1) Instance, Schema
2) Relation, Schema
3) Relation, Domain
4) Schema, Instance

Answer is: 4

Instance is an instance of time and schema is a representation.

Q10) Which option only works for a specialization whose subclasses are total (every entity in the superclass must belong to (at least) one of the subclasses) while the mapping of specialisation / generalisation form EER diagram to relational table:

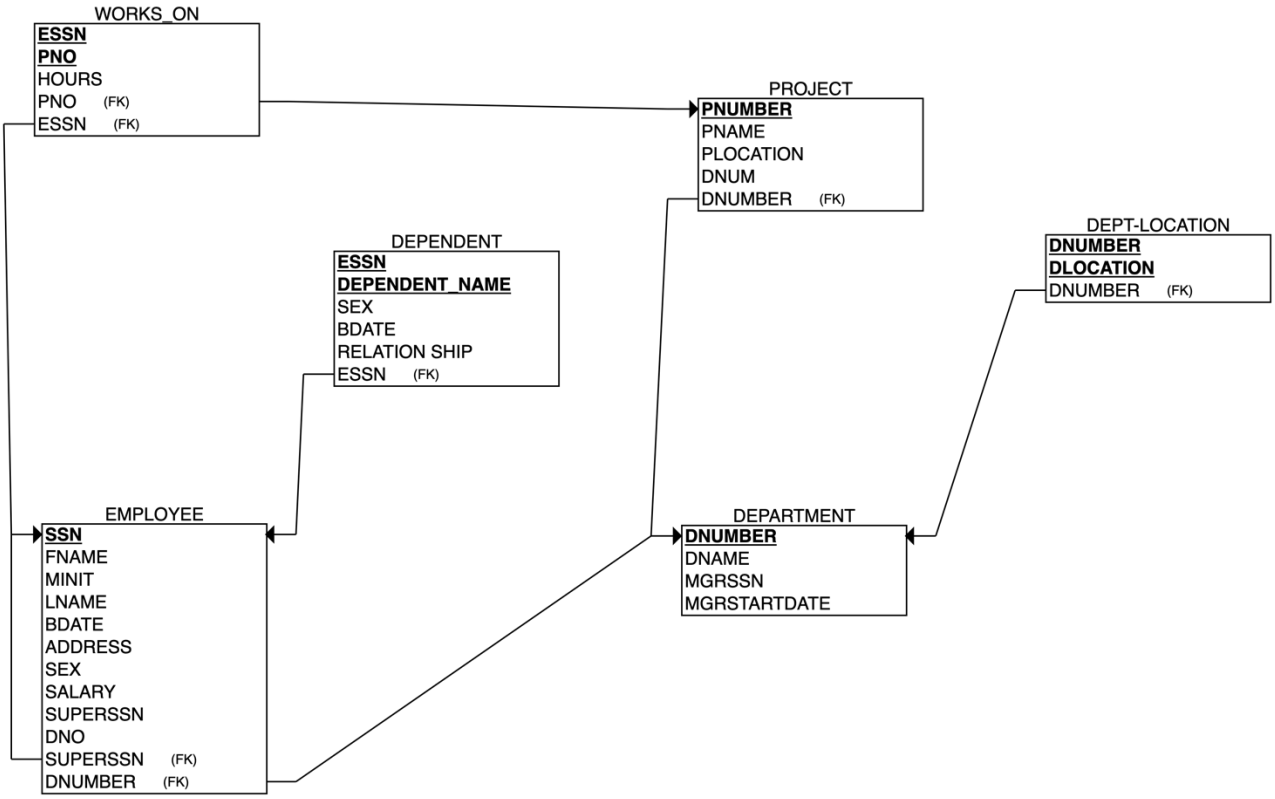
1) Multiple relations -Superclass and subclasses
2) Multiple relations -Subclass relations only
3) Single relation with *one type attribute*
4) Single relation with multiple type attributes.

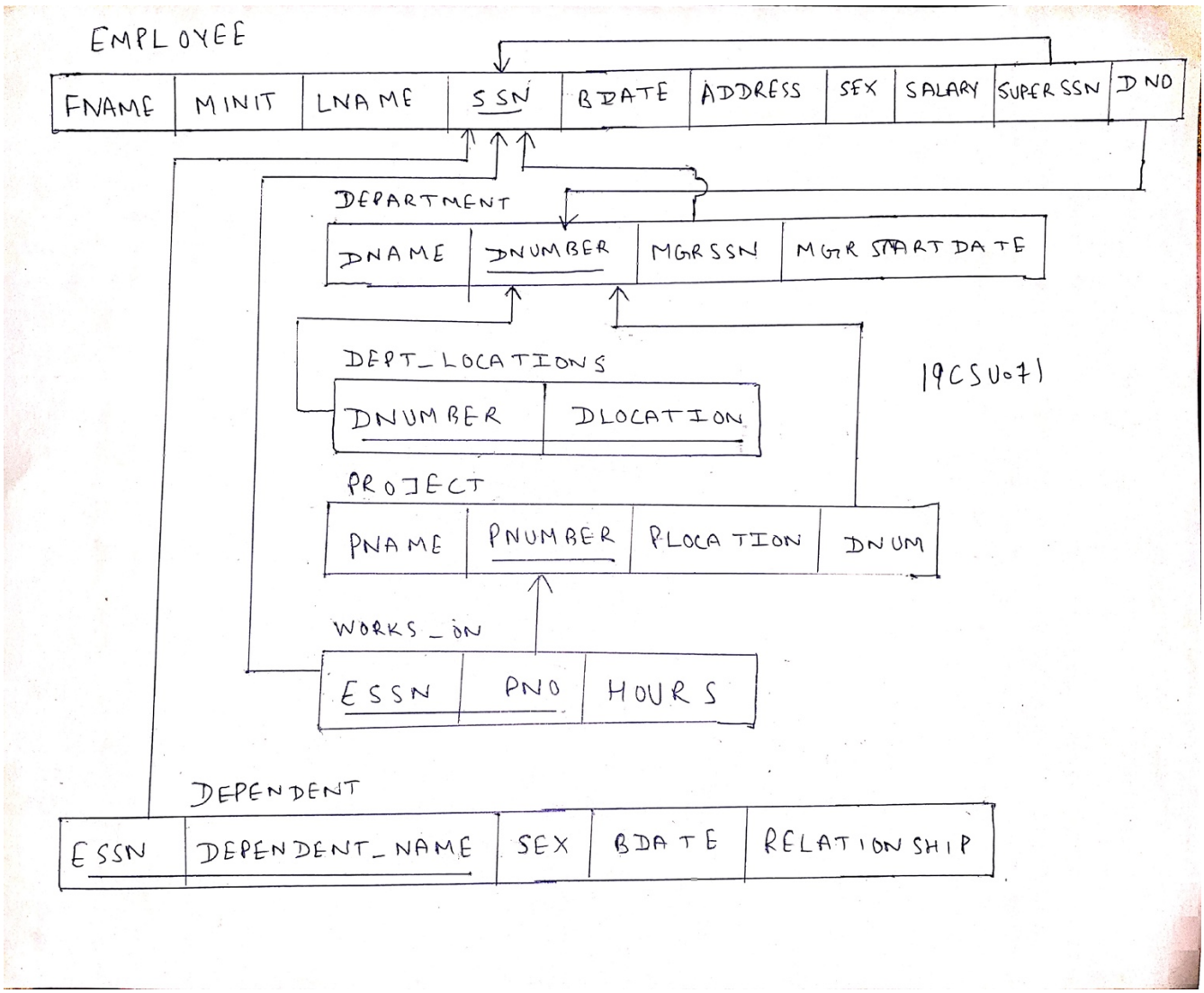
Answer is: 2

Student Work Area

Algorithm/Flowchart/Code/Sample Outputs

Relational Database Design




EMPLOYEE
FNAME
MINIT
LNAME
SSN
BDATE
ADDRESS
SEX
SALARY
SUPERSSN
DNO
DEPARTMENT
DNAME
DNUMBER
MGRSSN
MGRSTARTDATE
DEPT_LOCATIONS
DNUMBER
DLOCATION
19CSU071
PROJECT
PNAME
PNUMBER
PLOCATION
DNUM
WORKS_ON
ESSN
PNO
HOURS
DEPENDENT
ESSN
DEPENDENT_NAME
SEX
BDATE
RELATIONSHIP

EXPERIMENT NO. 3

Student Name and Roll Number: Chirag Sardana and 19CSU071
Semester /Section: 4th/ FS A1
Link to Code: https://github.com/chiragsardana/Question-In-Java/tree/master/DBMS/Experiment%203
Date:
Faculty Signature:
Marks:

Objective(s):

To apply SQL integrity constraints as per the DDL statements given below for COMPANY database.

Outcome:

- The students will understand how a database is created followed by insertion of relevant data.
- The students will understand the need of applying various types of integrity constraints such as primary key, foreign key, unique key, NOT NULL, default and CHECK etc

Problem Statement:

```
CREATE TABLE EMPLOYEE
        ( Fname          VARCHAR(15)          NOT NULL,
          Minit          CHAR,
          Lname          VARCHAR(15)          NOT NULL,
          Ssn            CHAR(9)              NOT NULL,
          Bdate          DATE,
          Address        VARCHAR(30),
          Sex            CHAR,
          Salary         DECIMAL(10,2),
          Super_ssn      CHAR(9),
          Dno            INT                  NOT NULL,
        PRIMARY KEY (Ssn),
        FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE(Ssn),
        FOREIGN KEY (Dno) REFERENCES DEPARTMENT(Dnumber) );
```

```
CREATE TABLE DEPARTMENT
        ( Dname                 VARCHAR(15)             NOT NULL,
          Dnumber               INT                     NOT NULL,
          Mgr_ssn               CHAR(9)                 NOT NULL,
          Mgr_start_date        DATE,
        PRIMARY KEY (Dnumber),
        UNIQUE (Dname),
        FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn) );

CREATE TABLE DEPT_LOCATIONS
        ( Dnumber               INT                     NOT NULL,
          Dlocation             VARCHAR(15)             NOT NULL,
        PRIMARY KEY (Dnumber, Dlocation),
        FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber) );
CREATE TABLE PROJECT
        ( Pname                 VARCHAR(15)             NOT NULL,
          Pnumber               INT                     NOT NULL,
          Plocation             VARCHAR(15),
          Dnum                  INT                     NOT NULL,
        PRIMARY KEY (Pnumber),
        UNIQUE (Pname),
        FOREIGN KEY (Dnum) REFERENCES DEPARTMENT(Dnumber) );
CREATE TABLE WORKS_ON
        ( Essn                  CHAR(9)                 NOT NULL,
          Pno                   INT                     NOT NULL,
          Hours                 DECIMAL(3,1)            NOT NULL,
        PRIMARY KEY (Essn, Pno),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn),
        FOREIGN KEY (Pno) REFERENCES PROJECT(Pnumber) );
CREATE TABLE DEPENDENT
        ( Essn                  CHAR(9)                 NOT NULL,
          Dependent_name        VARCHAR(15)             NOT NULL,
          Sex                   CHAR,
          Bdate                 DATE,
          Relationship          VARCHAR(8),
        PRIMARY KEY (Essn, Dependent_name),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn) );
```

Implement the following types of integrity constraints:

1) Primary Key
2) Foreign Key
3) Unique
4) Default
5) Auto-increment
6) Check
7) Not Null

Background Study:

1) Primary Key Constraint: A column or group of columns in a table which helps us to uniquely identifies every row in that table is called a primary key. This DBMS can't be a duplicate. The same value can't appear more than once in the table.

Syntax to define a Primary key at column level:
column name datatype [CONSTRAINT constraint_name] PRIMARY KEY

Syntax to define a Primary key at table level:
[CONSTRAINT constraint_name] PRIMARY KEY (column_name1,column_name2,..)

Rules for defining Primary key:

- Two rows can't have the same primary key value
- It must for every row to have a primary key value.
- The primary key field cannot be null.
- The value in a primary key column can never be modified or updated if any foreign key refers to that primary key.

2) Foreign Key (Referential integrity constraint): This constraint identifies any column referencing the PRIMARY KEY in another table. It establishes a relationship between two columns in the same table or between different tables. For a column to be defined as a Foreign Key, it should be a defined as a Primary Key in the table which it is referring. One or more columns can be defined as Foreign key.

Syntax to define a Foreign key at column level:
[CONSTRAINT constraint_name] REFERENCES Referenced_Table_name(column_name)

Syntax to define a Foreign key at table level:
[CONSTRAINT constraint_name] FOREIGN KEY(column_name) REFERENCES referenced_table_name(column_name);

3) SQL Not Null Constraint : This constraint ensures all rows in the table contain a definite value for the column which is specified as not null. Which means a null value is not allowed.

Syntax to define a Not Null constraint:
[CONSTRAINT constraint name] NOT NULL

4) **SQL Unique Key:** This constraint ensures that a column or a group of columns in each row have a distinct value. A column(s) can have a null value but the values cannot be duplicated.

Syntax to define a Unique key at column level:
[CONSTRAINT constraint_name] UNIQUE

Syntax to define a Unique key at table level:
[CONSTRAINT constraint_name] UNIQUE(column_name)

5) **SQL Check Constraint :** This constraint defines a business rule on a column. All the rows must satisfy this rule. The constraint can be applied for a single column or a group of columns.
Syntax to define a Check constraint:
[CONSTRAINT constraint_name] CHECK (condition)

Question Bank:

1) Suppose (A, B) and (C,D) are two relation schemas. Let r1 and r2 be the corresponding relation instances. B is a foreign key that refers to C in r2. If data in r1 and r2 satisfy referential integrity constraints, which of the following is ALWAYS TRUE?

(A) $\Pi_B(r_1) - \Pi_C(r_2) = \varnothing$

(B) $\Pi_C(r_2) - \Pi_B(r_1) = \varnothing$

(C) $\Pi_B(r_1) = \Pi_C(r_2)$

(D) $\Pi_B(r_1) - \Pi_C(r_2) \neq \varnothing$

1) A
2) B
3) C
4) D

Answer is: 1

B is a foreign key in r1 that refers to C in r2. r1 and r2 satisfy referential integrity constraints. So every value that exists in column B of r1 must also exist in column C of r2.

2) Given the following statements:
S1: A foreign key declaration can always be replaced by an equivalent check assertion in SQL.
S2: Given the table R (a, b, c) where a and b together form the primary key, the following is a valid table definition.

```
CREATE TABLE S (
        a INTEGER,
        d INTEGER,
        e INTEGER,
        PRIMARY KEY (d),
        FOREIGN KEY (a) references R)
```

Which one of the following statements is CORRECT?

1) S1 is TRUE and S2 is FALSE.
2) Both S1 and S2 are TRUE.
3) S1 is FALSE and S2 is TRUE.
4) Both S1 and S2 are FALSE

Answer is: 4

Check assertions are not sufficient to replace foreign key. Foreign key declaration may have cascade delete which is not possible by just check insertion.

Using a check condition we can have the same effect as Foreign key while adding elements to the child table. But when we delete an element from the parent table the referential integrity constraint is no longer valid. So, a check constraint cannot replace a foreign key.

So, we cannot replace it with a single check.

Foreign key in one table should uniquely identifies a row of other table. In above table definition, table S has a foreign key that refers to field 'a' of R. The field 'a' in table S doesn't uniquely identify a row in table R.

Q3) Which of the following is not an integrity constraint?

1) Not null
2) Positive
3) Unique
4) Check 'predicate'

Answer is: 2

Positive is a value and not a constraint.

Q4) *CREATE TABLE Manager(ID NUMERIC,Name VARCHAR(20),budget NUMERIC,Details VARCHAR(30));*

In order to ensure that the value of budget is non-negative which of the following should be used?

1)Check(budget>0)
2)Check(budget<0)
3)Alter(budget>0)
4) Alter(budget<0)

Answer is: 1

A common use of the check clause is to ensure that attribute values satisfy specified conditions, in effect creating a powerful type system.

Q5) The following table has two attributes A and C where A is the primary key and C is the foreign key referencing A with on-delete cascade.

A	C
2	4
3	4
4	3
5	2
7	2
9	5
6	4

The set of all tuples that must be additionally deleted to preserve referential integrity when the tuple (2,4) is deleted is:

1) (3,4) and (6,4)
2) (5,2) and (7,2)

3) (5,2), (7,2) and (9,5)
4) (3,4), (4,3) and (6,4)

Answer is: 3

When (2,4) is deleted. Since C is a foreign key referring A with delete on cascade, all entries with value 2 in C must be deleted. So (5, 2) and (7, 2) are deleted. As a result of this 5 and 7 are deleted from A which causes (9, 5) to be deleted.

Q6) *CREATE TABLE course*
(. . .
FOREIGN KEY (dept name) REFERENCES department. . .);
Which of the following is used to delete the entries in the referenced table when the tuple is deleted in course table?

1) Delete
2) Delete cascade
3) Set null
4) All of the mentioned

Answer is: 2

Q7) Domain constraints, functional dependency and referential integrity are special forms of ___

1) Foreign key
2) Primary key
3) Assertion
4) Referential constraint

Answer is: 3

Q8) Which of the following can be addressed by enforcing a referential integrity constraint?

1) All phone numbers must include the area code
2) Certain fields are required (such as the email address, or phone number) before the record is accepted
3) Information on the customer must be known before anything can be sold to that customer

4) When entering an order quantity, the user must input a number and not some text (i.e., 12 rather than 'a dozen')

Answer is: 3

The information can be referred to and obtained.

Q9) Given relations r (w, x) and s(y, z), the result of

SELECT DISTINCT w, x

FROM r, s

is guaranteed to be same as r, provided

1) r has no duplicates and s is non-empty
2) r and s have no duplicates
3) s has no duplicates and r is non-empty
4) r and s have the same number of tuples

Answer is: 1

The query selects all attributes of r. Since we have distinct in query, result can be equal to r only if r doesn't have duplicates.

If we do not give any attribute on which we want to join two tables, then the queries like above become equivalent to Cartesian product. Cartisian product of two sets will be empty if any of the two sets is empty. So, s should have atleast one record to get all rows of r.

Q10) In SQL, relations can contain null values, and comparisons with null values are treated as unknown. Suppose all comparisons with a null value are treated as false. Which of the following pairs is not equivalent?

1) x = 5 AND not (not(x = 5))
2) x = 5 AND x> 4 and x < 6, where x is an integer
3) x ≠ 5 AND not (x = 5)
4) None of the above

Answer is: 3

According to given question, comparison with NULL value always False, so "x ≠ 5" will be

false. "x = 5" will also false and not (x = 5) = not(false) = true.

Hence, these are not equivalent pair.

Student Work Area

Algorithm/Flowchart/Code/Sample Outputs

Work:

Creating Table

1. Department Table:

```
create table DEPARTMENT
(
Dname varchar(15) not null,
Dnumber int not null,
Mgr_ssn char(9) not null,
Mgr_start_date date,
primary key(Dnumber),
unique(Dname)
# foreign key(Mgr_ssn) references EMPLOYEE(Ssn)
);
```

2. Employee Table:

```
create table EMPLOYEE
(
Fname varchar(15) not null,
Minit char,
Lname varchar(15) not null,
Ssn char(9) not null,
Bdate date,
Address varchar(30),
Sex char,
Salary decimal(10,2),
Super_ssn char(9),
Dno int not null,
primary key(Ssn),
foreign key(Super_ssn) references EMPLOYEE(Ssn),
foreign key(Dno) references DEPARTMENT(Dnumber)
);
```

3. Dept_Locations Table:

```
create table DEPT_LOCATIONS
```

```
(
Dnumber int not null,
Dlocation varchar(15),
PRIMARY KEY (Dnumber,Dlocation),
FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber)
);
```

4. Project Table:

```
create TABLE PROJECT
(
Pname varchar(15) not null,
Pnumber int not null,
Plocation varchar(15),
Dnum int not null ,
primary key (Pnumber),
UNIQUE(Pname),
FOREIGN KEY(Dnum) references DEPARTMENT(Dnumber)
);
```

5. Works_On Table:

```
create table WORKS_ON
(
Essn char(9) not null,
Pno int not null,
Hours DECIMAL(3,1) not null,
PRIMARY KEY(Essn,Pno),
FOREIGN KEY(Essn) references EMPLOYEE(Ssn),
FOREIGN KEY (Pno) references PROJECT(Pnumber)
);
```

6. Dependent Table:

```
CREATE table DEPENDENT
(
Essn char(9) not null ,
Dependent_name varchar(15) not null ,
Sex char,
Bdate DATE,
Relationship varchar(8),
PRIMARY KEY(Essn,Dependent_name),
```

```
FOREIGN KEY(Essn) references EMPLOYEE(Ssn)

);
```

Inserting Values in the Tables

1. Inserting Values in DepartmentTable:

```
INSERT INTO DEPARTMENT VALUES ('CSE', 1, '123456789', '2020-01-15');

INSERT INTO DEPARTMENT (Dname, Dnumber, Mgr_ssn)

VALUES ('ECE', 2, '123456780');

INSERT INTO DEPARTMENT VALUES ('IT', 3, '123456781', '2020-10-14');

INSERT INTO DEPARTMENT (Dname, Dnumber, Mgr_ssn, Mgr_start_date)

VALUES ('CSE-Full Stack', 4, '123456782', '2021-01-12'),

    ('CSE-Cyber',5,'123456784', '2020-10-13');
```

2. Inserting Values in Employee Table:

```
INSERT INTO EMPLOYEE (Fname, Lname, Ssn, Bdate, Address, Sex, Salary, Dno)
```

VALUES ('Chirag', 'Sardana', '123456782', '2000-10-13', 'Sirsa', 'M', 10000000.12,1);

INSERT INTO EMPLOYEE (Fname, Minit, Lname, Ssn, Bdate, Address, Sex, Salary, Super_ssn, Dno)

VALUES ('Anuj', ' ', 'Sharma', '123456783', '2001-10-12', 'Delhi', 'M', 12000000.12, '123456782',1);

INSERT INTO EMPLOYEE (Fname, Lname, Ssn, Bdate, Address, Sex, Salary, Dno)

VALUES ('Ayush', 'Singh', '123456784', '2001-05-05', 'Delhi', 'M', 12000000.12,1);

INSERT INTO EMPLOYEE (Fname, Minit, Lname, Ssn, Bdate, Address, Sex, Salary, Super_ssn, Dno)

VALUES ('Deepak',' ', 'Jindal', '123456785', '2000-08-26','Near DTU Delhi','M',10000000.12,'123456782',1),

('Namit', ' ', 'Kumar','123456786', '2000-08-26', 'Gurugram' , 'M', 10000000.12, '123456785',2);

3. Inserting Values in Dept_Locations Table:

INSERT INTO DEPT_LOCATIONS (Dnumber, Dlocation)

VALUES (1, 'Gurugram');

INSERT INTO DEPT_LOCATIONS (Dnumber, Dlocation)

VALUES (1, 'Delhi');

```
INSERT INTO DEPT_LOCATIONS (Dnumber, Dlocation)

VALUES (1, 'Sirsa');
```

4. Inserting Values in Project Table:

```
INSERT INTO PROJECT (Pname, Pnumber, Plocation, Dnum)

VALUES ('getRide', 71, 'Gurugram', 1);


INSERT INTO PROJECT (Pname, Pnumber, Dnum)

VALUES ('Light', 70, 2);


INSERT INTO PROJECT (Pname, Pnumber, Plocation, Dnum)

VALUES ('PortFolio', 711, 'NCU', 3);
```

5. Inserting Values in Works_On Table:

```
INSERT INTO WORKS_ON (Essn, Pno, Hours)

VALUES ('123456782', 71, 12.1);


INSERT INTO WORKS_ON (Essn, Pno, Hours)

VALUES ('123456783', 711, 11.2);
```

INSERT INTO WORKS_ON (Essn, Pno, Hours)

VALUES ('123456784', 70, 24.2);

6. Inserting Values in Dependent Table:

INSERT INTO DEPENDENT (Essn, Dependent_name, Sex, Bdate, Relationship)

VALUES ('123456782', 'Dheeraj Sardana','M', '2000-12-10', 'Brother');

INSERT INTO DEPENDENT (Essn, Dependent_name, Sex, Bdate, Relationship)

VALUES ('123456782', 'Dheeraj Sardana','M', '2000-12-10', 'Brother');

INSERT INTO DEPENDENT (Essn, Dependent_name)

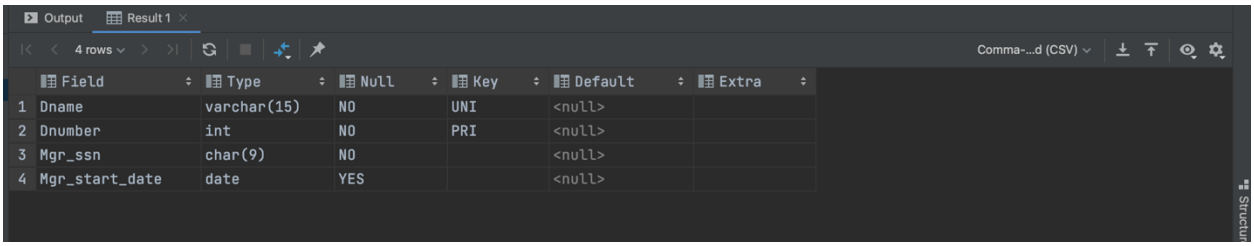
VALUES ('123456783', 'Anuj Sharma');

INSERT INTO DEPENDENT (Essn, Dependent_name, Relationship)

VALUES ('123456782', 'Deepak','Friend');

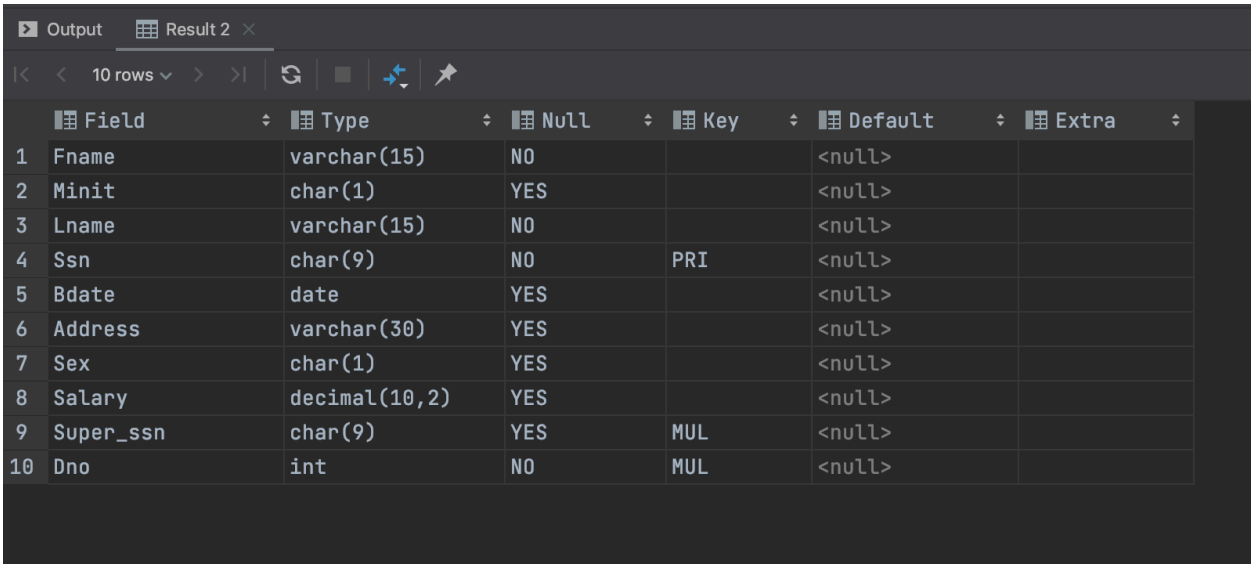
Output: Screenshots

Command: DESC DEPARTMENT;



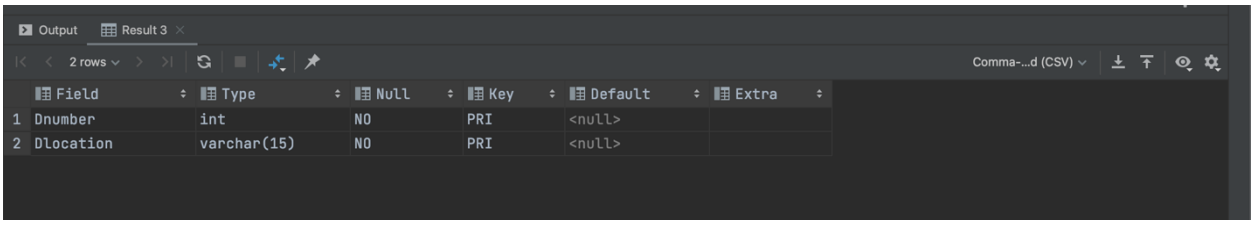
	Field	Type	Null	Key	Default	Extra
1	Dname	varchar(15)	NO	UNI	<null>	
2	Dnumber	int	NO	PRI	<null>	
3	Mgr_ssn	char(9)	NO		<null>	
4	Mgr_start_date	date	YES		<null>	

Command: DESC EMPLOYEE;



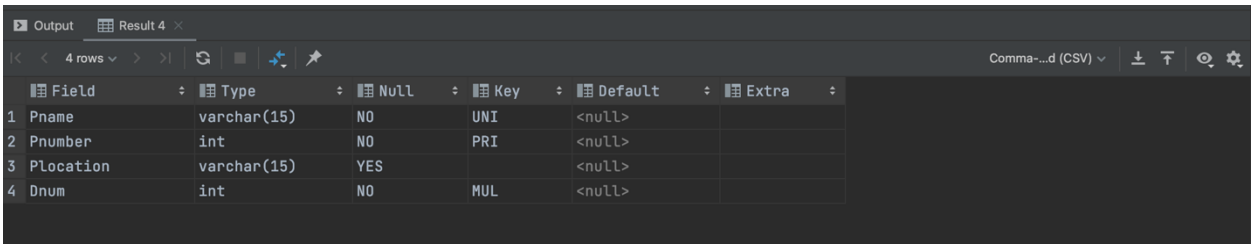
	Field	Type	Null	Key	Default	Extra
1	Fname	varchar(15)	NO		<null>	
2	Minit	char(1)	YES		<null>	
3	Lname	varchar(15)	NO		<null>	
4	Ssn	char(9)	NO	PRI	<null>	
5	Bdate	date	YES		<null>	
6	Address	varchar(30)	YES		<null>	
7	Sex	char(1)	YES		<null>	
8	Salary	decimal(10,2)	YES		<null>	
9	Super_ssn	char(9)	YES	MUL	<null>	
10	Dno	int	NO	MUL	<null>	

Command: DESC DEPT_LOCATIONS;



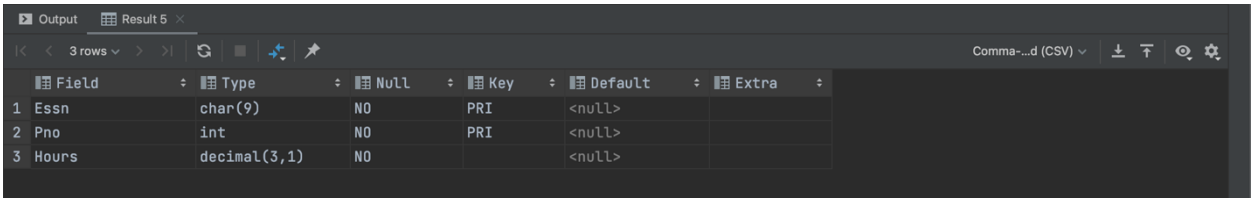
	Field	Type	Null	Key	Default	Extra
1	Dnumber	int	NO	PRI	<null>	
2	Dlocation	varchar(15)	NO	PRI	<null>	

Command: DESC PROJECT;



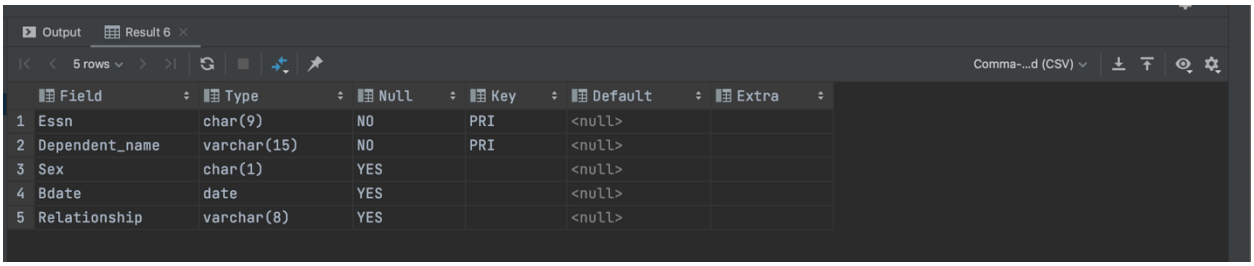
	Field	Type	Null	Key	Default	Extra
1	Pname	varchar(15)	NO	UNI	<null>	
2	Pnumber	int	NO	PRI	<null>	
3	Plocation	varchar(15)	YES		<null>	
4	Dnum	int	NO	MUL	<null>	

Command: DESC WORKS_ON;



	Field	Type	Null	Key	Default	Extra
1	Essn	char(9)	NO	PRI	<null>	
2	Pno	int	NO	PRI	<null>	
3	Hours	decimal(3,1)	NO		<null>	

Command: DESC DEPENDENT;



	Field	Type	Null	Key	Default	Extra
1	Essn	char(9)	NO	PRI	<null>	
2	Dependent_name	varchar(15)	NO	PRI	<null>	
3	Sex	char(1)	YES		<null>	
4	Bdate	date	YES		<null>	
5	Relationship	varchar(8)	YES		<null>	

Command: SELECT * FROM DEPARTMENT;

Output NCU.DEPARTMENT

5 rows | Tx: Auto | DDL | DML | Comma-...d (CSV)

	Dname	Dnumber	Mgr_ssn	Mgr_start_date
1	CSE	1	123456789	2020-01-15
2	ECE	2	123456780	<null>
3	IT	3	123456781	2020-10-14
4	CSE-Full Stack	4	123456782	2021-01-12
5	CSE-Cyber	5	123456784	2020-10-13

Command: SELECT * FROM EMPLOYEE;

Output NCU.EMPLOYEE

5 rows | Tx: Auto | DDL | DML | Comma-...d (CSV)

	Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
1	Chirag	<null>	Sardana	123456782	2000-10-13	Sirsa	M	10000000.12	<null>	1
2	Anuj		Sharma	123456783	2001-10-12	Delhi	M	12000000.12	123456782	1
3	Ayush	<null>	Singh	123456784	2001-05-05	Delhi	M	12000000.12	<null>	1
4	Deepak		Jindal	123456785	2000-08-26	Near DTU Delhi	M	10000000.12	123456782	1
5	Namit		Kumar	123456786	2000-08-26	Gurugram	M	10000000.12	123456785	2

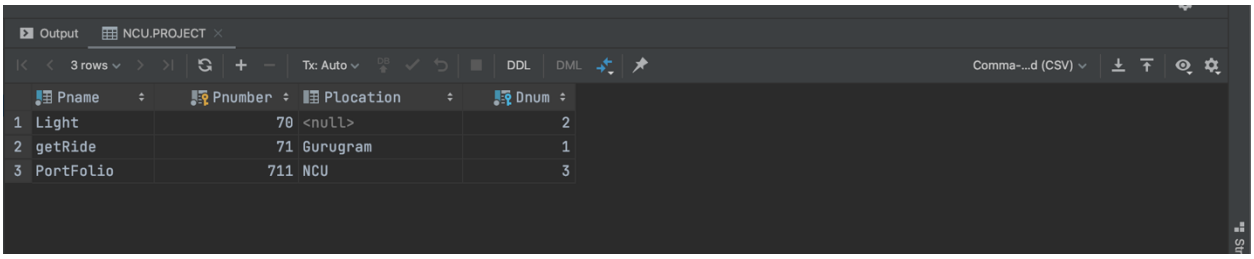
Command: SELECT * FROM DEPT_LOCATIONS;

Output NCU.DEPT_LOCATIONS

3 rows | Tx: Auto | DDL | DML | Comma-...d (CSV)

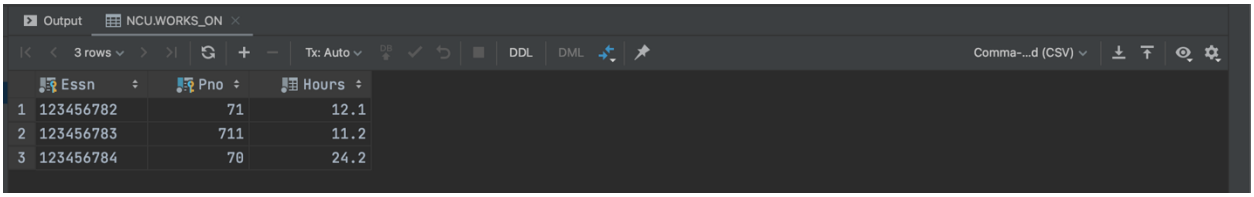
	Dnumber	Dlocation
1	1	Delhi
2	1	Gurugram
3	1	Sirsa

Command: SELECT * FROM PROJECT;



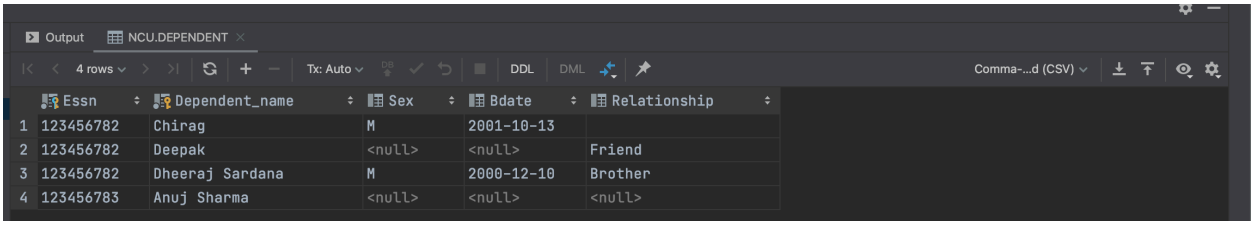
	Pname	Pnumber	Plocation	Dnum
1	Light	70	<null>	2
2	getRide	71	Gurugram	1
3	PortFolio	711	NCU	3

Command: SELECT * FROM WORKS_ON;



	Essn	Pno	Hours
1	123456782	71	12.1
2	123456783	711	11.2
3	123456784	70	24.2

Command: SELECT * FROM DEPENDENT;



	Essn	Dependent_name	Sex	Bdate	Relationship
1	123456782	Chirag	M	2001-10-13	
2	123456782	Deepak	<null>	<null>	Friend
3	123456782	Dheeraj Sardana	M	2000-12-10	Brother
4	123456783	Anuj Sharma	<null>	<null>	<null>

EXPERIMENT NO. 4

Student Name and Roll Number: Chirag Sardana and 19CSU071
Semester /Section: 4th/ FS A1
Link to Code: https://github.com/chiragsardana/Question-In-Java/tree/master/DBMS/Experiment%204
Date:
Faculty Signature:
Marks:

Objective(s):

To familiarize with SELECT-FROM-WHERE SQL simple queries on the COMPANY database.

Outcome:

The students will be able to retrieve zero or more rows from one or more database tables or database views.

Problem Statement:

From the COMPANY database as mentioned and described in the previous database :

1) Retrieve the birth date and address of the employee(s) whose name is 'John B. Smith'.
2) Retrieve the name and address of all employees who work for the 'Research' department.
3) For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, address, and birth date
4) Select all combinations of EMPLOYEE Ssn and DEPARTMENT Dname in the database.
5) Retrieve all the attribute values of any EMPLOYEE who works in DEPARTMENT number 5
6) Retrieve all distinct salary values.
7) Make a list of all project numbers for projects that involve an employee whose last name is 'Smith', either as a worker or as a manager of the department that controls the project.
8) Retrieve all employees whose address is in Houston, Texas.
9) Find all employees who were born during the 1950s
10) Show the resulting salaries if every employee working on the 'ProductX' project is given a 10 percent raise

11) Retrieve a list of employees and the projects they are working on, ordered by department and, within each department, ordered alphabetically by last name, then first name.

Background Study:

Structured Query Language SQL contains statements for data definitions, queries, and updates (both DDL and DML)

1) Domain
   - Name used with the attribute specification
   - Makes it easier to change the data type for a domain that is used by numerous attributes

2) Inserting values in our table using the commands :
   INSERT INTO EMPLOYEE (Fname, Lname, Dno, Ssn)

   VALUES ('Richard', 'Marini', 4, '653298653')

3) Out of the complete Database we create we can easily pick out and filter certain amount of data by using the SQL queries as :

   SELECT <attribute list>

   FROM <table list>

   WHERE <condition>;

   - <attribute list> is a list of attribute names whose values are to be retrieved by the query.
   - <table list> is a list of the relation names required to process the query.
   - <condition> is a conditional (Boolean) expression that identifies the tuples to be retrieved by the query.

4) Ambiguous Attribute Names : Same name can be used for two (or more) attributes as long as the attributes are in different relations, these are made to differ by mentioning their table names before the attribute names.
5) Aliasing, Renaming Tuple variables : We can rename the tables into some smaller easier names as per choice when it has to be used multiple times in the query.

6) DISTINCT keyword is used when the query wishes to derive no two duplicate values.

7) SET operations such as UNION and INTERSECT can also be applied on the tables to filter out the values as per choice.

8) Substring Pattern matching is carried out by the use of the keyword LIKE which helps in retrieving the column values of the tuple matching our mentioned substring.

9) ORDERBY <attributes>
   - Helps in sorting the tuple values of the attributes by default set to ascending and can be changed to descending .

Question Bank:

Q1) Which operator performs Pattern matching ?

1) BETWEEN operator
2) LIKE operator
3) EXISTS operator
4) None of the above

Answer is: 2

LIKE is a keyword that is used in the WHERE clause. Basically, LIKE allows us to do a search based operation on a pattern rather than specifying exactly what is desired (as in IN) or spell out a range (as in BETWEEN).

Q2) In SQL which commands are used to change the storage characteristics of the table?

1) ALTER TABLE
2) MODIFY TABLE
3) CHANGE TABLE
4) All of the above

Answer is: 1

To change the structure of the table we use ALTER TABLE.

Syntax:

```
ALTER TABLE "table_name"
ADD "column_name" datatype
```

Q3) _____removes all rows from a table without logging the individual row deletions.

1) DELETE
2) REMOVE
3) DROP
4) TRUNCATE

Answer is: 4

TRUNCATE statement is a Data Definition Language (DDL) operation that marks the extents of a table for deallocation.

Q4) If you don't specify ASC or DESC after a SQL ORDER BY clause, the following is used by default :

1) ASC
2) DESC
3) There is no default value
4) None of the mentioned

Answer is: 1

Q5) What is the purpose of the SQL AS clause?

1) The AS SQL clause is used to change the name of a column in the result set or to assign a name to a derived column
2) The AS clause is used with the JOIN clause only
3) The AS clause defines a search condition
4) All of the mentioned

Answer is: 1

SQL AS clauses are defined for columns and tables to give an alias name. Basically, aliases are

created to increase the readability of the query and column headings in the output.

Q6) Which of the following command is used to delete a table in SQL?

1) Delete
2) Truncate
3) Remove
4) Drop

Answer is: 4

Q7) Which of the following commands makes the updates performed by the transaction permanent in the database?

1) ROLLBACK
2) COMMIT
3) TRUNCATE
4) DELETE

Answer is: 2

Q8) Select operation in SQL is equivalent to

1) the selection operation in relational algebra
2) the selection operation in relational algebra, except that select in SQL retains duplicates
3) the projection operation in relational algebra
4) the projection operation in relational algebra, except that select in SQL retains duplicates

Answer is: 4

Select operation is equivalent to the projection operation in relational algebra, except that select in SQL retains duplicates and on the contrary projection removes the duplicates

Student Work Area

Algorithm/Flowchart/Code/Sample Outputs

Work:

Before Retrieving, I am going to Insert Data More According to the Question Given Above.

```
INSERT INTO EMPLOYEE (Fname, Minit, Lname, Ssn, Bdate, Address, Sex, Salary, Dno)

VALUES ('John', "B", 'Smith', '12345677', '1990-05-05', 'Delhi', 'M', 12000000.12,1);

INSERT INTO DEPARTMENT VALUES ('Research', 6, '12345677', '2020-10-14');

INSERT INTO EMPLOYEE (Fname,Lname, Ssn, Bdate, Address, Sex, Salary, Dno)

VALUES ('Aastha', ' ','12345678', '1949-05-05', 'Delhi', 'F', 12000000.12,6);

INSERT INTO PROJECT (Pname, Pnumber, Plocation, Dnum)

VALUES ('LOS', 108, 'houston', 6);

INSERT INTO EMPLOYEE (Fname,Lname, Ssn, Bdate, Address, Sex, Salary, Dno)

VALUES ('Kanu', 'Sardana','12345679', '1950-05-05', 'Sirsa', 'M', 40000000.12,5);

INSERT INTO EMPLOYEE (Fname,Lname, Ssn, Bdate, Address, Sex, Salary, Dno)

VALUES ('Lovkesh', 'Sardana','123456712', '1950-10-13', 'Texas', 'M', 40000000.12,5);

INSERT INTO PROJECT (Pname, Pnumber, Plocation, Dnum)
```

VALUES ('ProductX', 121, 'Texas', 6);

INSERT INTO WORKS_ON (Essn, Pno, Hours)

VALUES ('123456712', 121, 12.2);

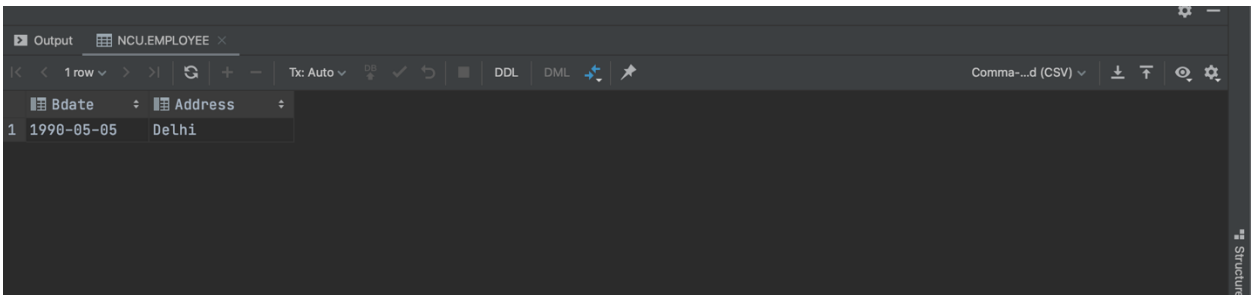
Output: Screenshots

From the COMPANY database as mentioned and described in the previous database :

1) Retrieve the birth date and address of the employee(s) whose name

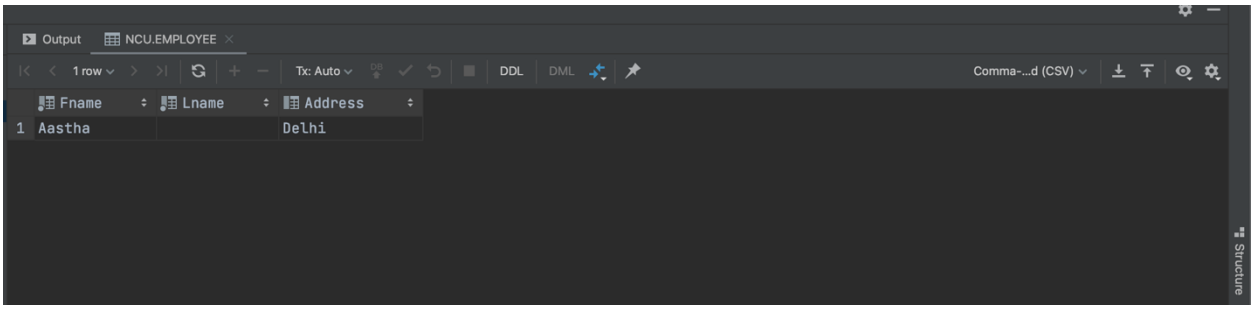
is ‘John B. Smith’.

Command: SELECT Bdate, Address FROM EMPLOYEE WHERE Fname= "john" AND Minit ="B" AND Lname ="Smith";



2) Retrieve the name and address of all employees who work for the ‘Research’ department.

Command: SELECT Fname, Lname, Address FROM EMPLOYEE, Department WHERE Dname="Research"AND Dnumber=Dno;

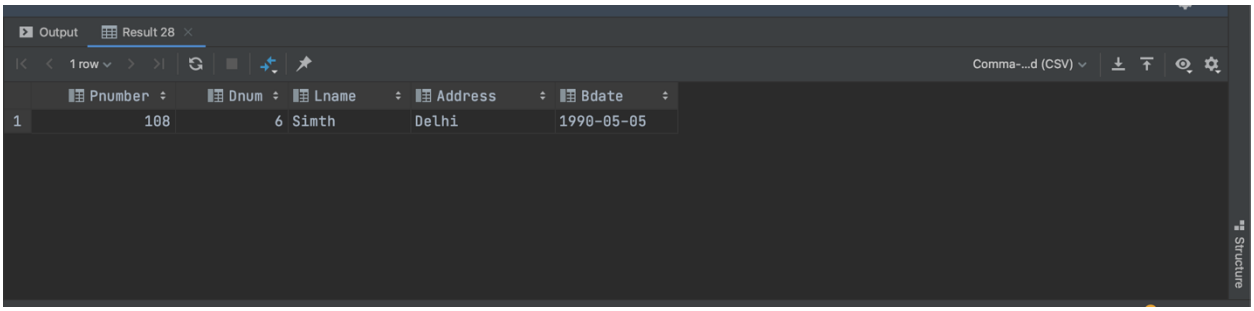


3) For every project located in 'houston', list the project number, the

controlling department number, and the department manager's last name,

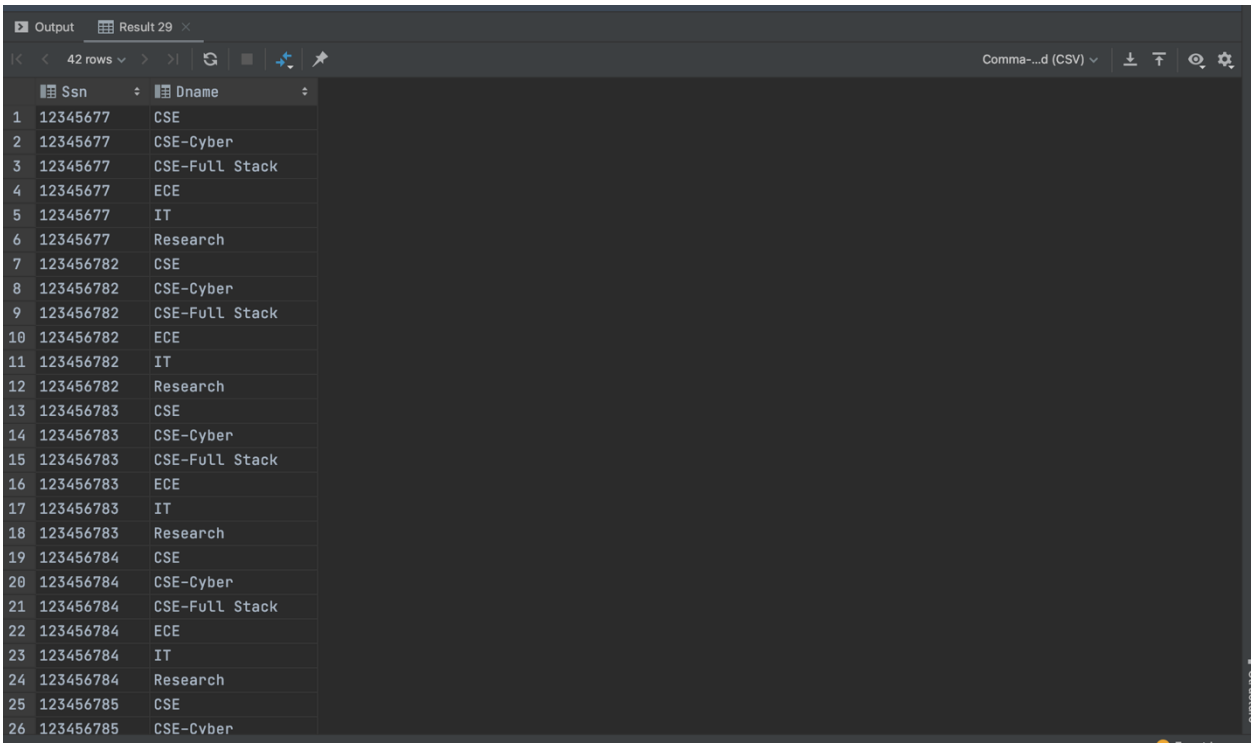
address, and birth date

Command: SELECT Pnumber,Dnum, Lname, Address, Bdate FROM PROJECT,DEPARTMENT,EMPLOYEE WHERE Dnum = Dnumber AND Mgr_ssn = Ssn AND Plocation="houston";

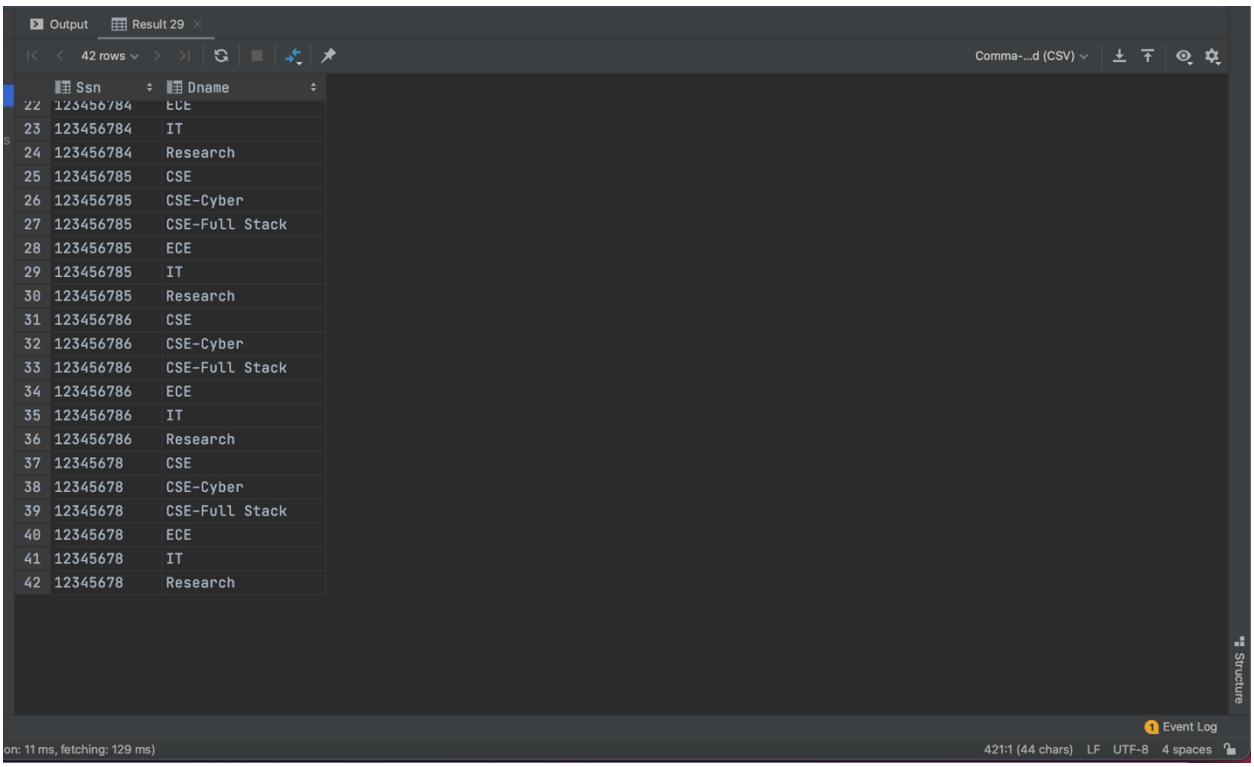


4) Select all combinations of EMPLOYEE Ssn and DEPARTMENT Dname in the database.

Command: SELECT Ssn, Dname FROM EMPLOYEE, Department;



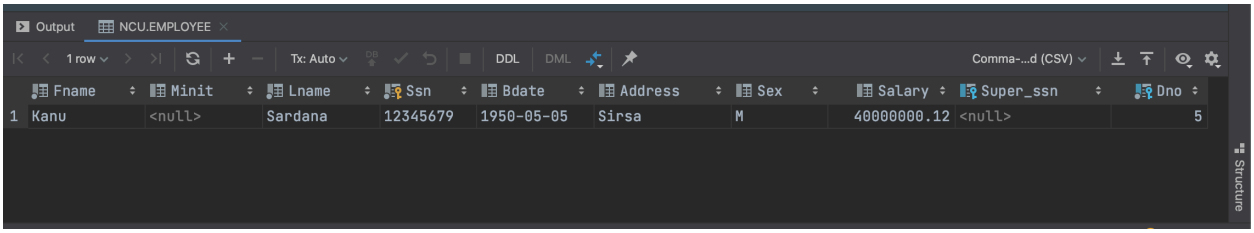
	Ssn	Dname
1	12345677	CSE
2	12345677	CSE-Cyber
3	12345677	CSE-Full Stack
4	12345677	ECE
5	12345677	IT
6	12345677	Research
7	123456782	CSE
8	123456782	CSE-Cyber
9	123456782	CSE-Full Stack
10	123456782	ECE
11	123456782	IT
12	123456782	Research
13	123456783	CSE
14	123456783	CSE-Cyber
15	123456783	CSE-Full Stack
16	123456783	ECE
17	123456783	IT
18	123456783	Research
19	123456784	CSE
20	123456784	CSE-Cyber
21	123456784	CSE-Full Stack
22	123456784	ECE
23	123456784	IT
24	123456784	Research
25	123456785	CSE
26	123456785	CSE-Cyber



	Ssn	Dname
22	123456784	ECE
23	123456784	IT
24	123456784	Research
25	123456785	CSE
26	123456785	CSE-Cyber
27	123456785	CSE-Full Stack
28	123456785	ECE
29	123456785	IT
30	123456785	Research
31	123456786	CSE
32	123456786	CSE-Cyber
33	123456786	CSE-Full Stack
34	123456786	ECE
35	123456786	IT
36	123456786	Research
37	12345678	CSE
38	12345678	CSE-Cyber
39	12345678	CSE-Full Stack
40	12345678	ECE
41	12345678	IT
42	12345678	Research

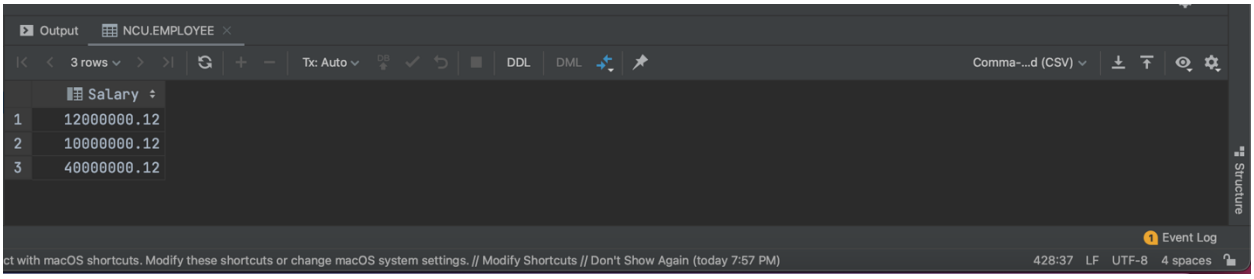
5) Retrieve all the attribute values of any EMPLOYEE who works in DEPARTMENT number 5

Command: SELECT * FROM EMPLOYEE WHERE Dno=5;



6) Retrieve all distinct salary values.

Command: SELECT DISTINCT Salary FROM EMPLOYEE;



7) Make a list of all project numbers for projects that involve an employee whose last name is

'Smith', either as a worker or as a manager of the department that controls the project.

Command: (SELECT DISTINCT(Pnumber)

FROM Project, Department, Employee

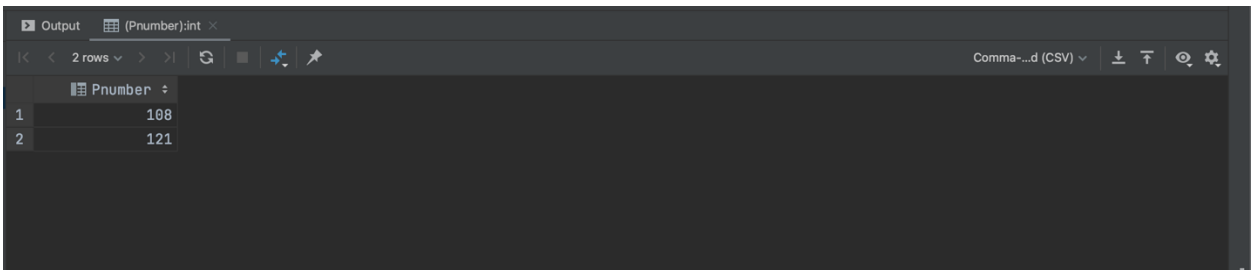
WHERE PROJECT.Dnum = DEPARTMENT.Dnumber AND Mgr_ssn = Ssn AND Lname='Smith')

UNION

(SELECT DISTINCT(Pnumber)

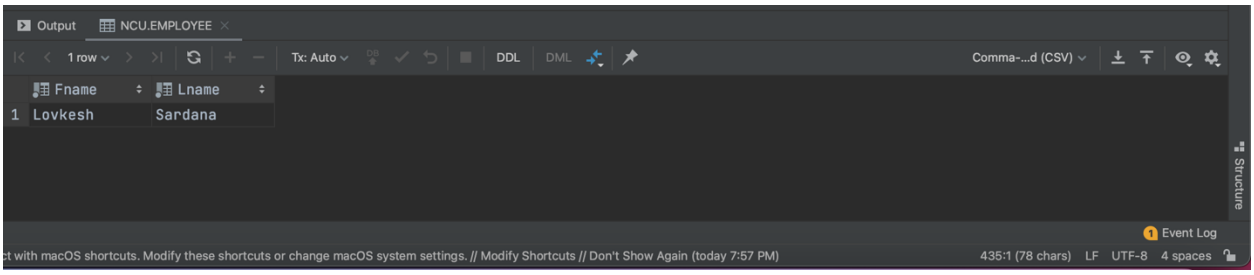
FROM Project, Works_On, Employee

WHERE PROJECT.Pnumber = WORKS_ON.Pno AND WORKS_ON.Essn=EMPLOYEE.Ssn AND Lname= 'Smith')



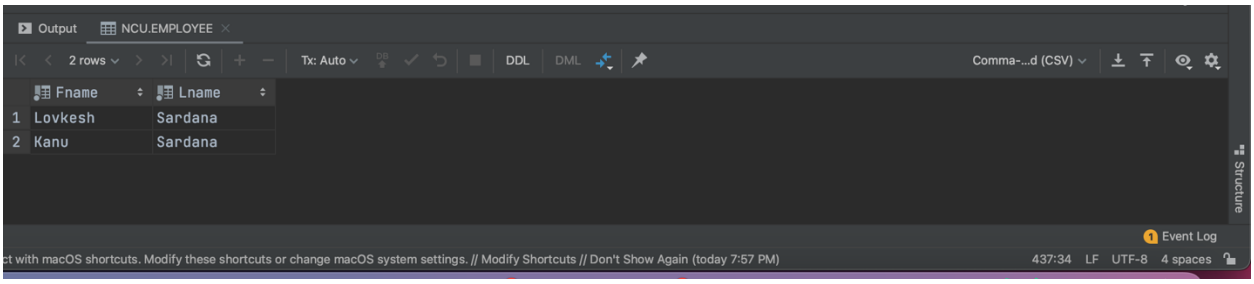
8) Retrieve all employees whose address is in Houston OR Texas.

Command: SELECT Fname, Lname FROM EMPLOYEE WHERE Address LIKE "%Texas%" OR "%Houstan%";



9) Find all employees who were born during the 1950s

Command: SELECT Fname, Lname FROM emp WHERE Bdate LIKE "1950%";



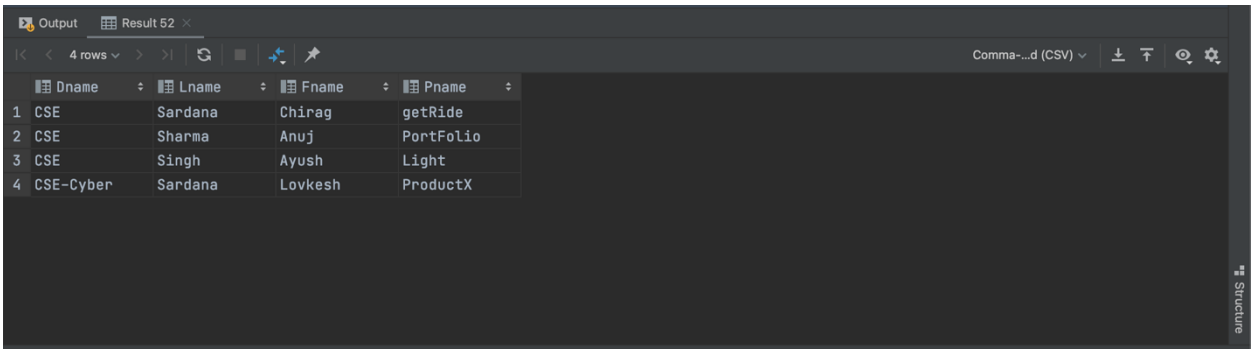
10) Retrieve a list of employees and the projects they are working on, ordered by department and, within each department, ordered alphabetically by last name, then first name.

Command: SELECT D.Dname, E.Lname, E.Fname, P.Pname

FROM Department AS D, Employee AS E, Works_On AS W, Project AS P

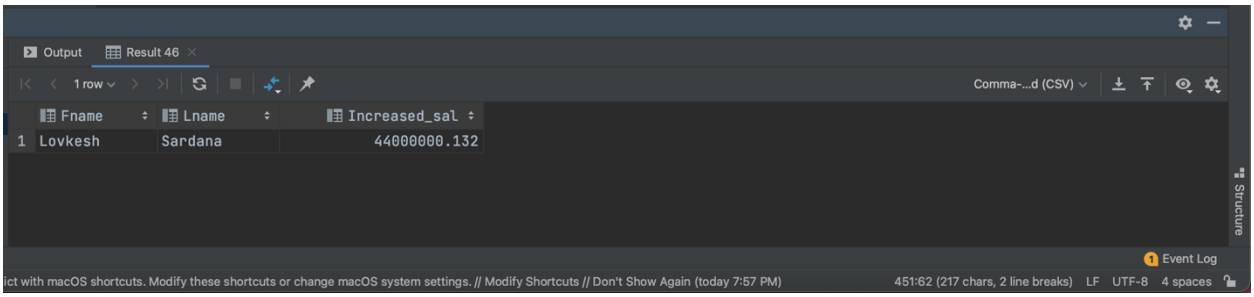
WHERE D.Dnumber=E.Dno AND E.Ssn=W.Essn AND W.Pno=P.Pnumber

ORDER BY D.Dname, E.Lname, E.Fname;



11)Show the resulting salaries if every employee working on the ‘ProductX’ project is given a 10 percent raise

Command: SELECT E.Fname, E.Lname, 1.1 * E.Salary AS Increased_sal FROM EMPLOYEE AS E, WORKS_ON AS W, PROJECT AS P WHERE E.Ssn=W.Essn AND W.Pno=P.Pnumber AND P.Pname="ProductX";



EXPERIMENT NO. 5

Student Name and Roll Number: Chirag Sardana and 19CSU071
Semester /Section: 4th/ FS A1
Link to Code: https://github.com/chiragsardana/Question-In-Java/tree/master/DBMS/Experiment%205
Date:
Faculty Signature:
Marks:

Objective(s):

To familiarize with different JOIN operations in SQL on the COMPANY database.

Outcome:

The students will be to establish a connection between two or more database tables based on matching columns, thereby creating a relationship between the tables.

Problem Statement:

Consider the Sample database given below and answer the queries as stated :

Pack_grades

Grade_id Grade_name Min_price Max_price

Customers

Customer_id First_name Last_name Birth_date Join_date City Pack_id State

Packages

Pack_id speed Start_date Monthly_payment Sector_id

Sectors

Sector_id Sector_name

1) Write a query to display first name, last name, package number and internet speed for all customers.
2) Write a query to display first name, last name, package number and internet speed for all customers whose package number equals 22 or 27. Order the query in ascending order by last name.
3) Display the package number, internet speed, monthly payment and sector name for all packages (*Packages* and *Sectors* tables).
4) Display the customer name, package number, internet speed, monthly payment and sector name for all customers (*Customers*, *Packages* and *Sectors* tables).
5) Display the customer name, package number, internet speed, monthly payment and sector name for all customers in the business sector (*Customers*, *Packages* and *Sectors* tables).
6) Display the last name, first name, join date, package number, internet speed and sector name for all customers in the private sector who joined the company in the year 2006.
7) Display the package number, internet speed, monthly payment and package grade for all packages (*Packages* and *Pack_Grades* tables).
8) Display the first name, last name, internet speed and monthly payment for all customers. Use INNER JOIN to solve this exercise.
9) Display the last name, first name and package number for all customers who have the same package number as customer named 'Amado Taylor' (*Customers* table).
10) Display the last name, first name and monthly discount for all customers whose monthly discount is lower than the monthly discount of employee number 103 (*Customers* table).
11) Display the package number and internet speed for all packages whose internet speed is equal to the internet speed of package number 10 (*Packages* table).

Background Study:

Aggregate Functions : Used to summarize information from multiple tuples into a single-tuple summary

1) Grouping : Create subgroups of tuples before summarizing
2) Built-in aggregate functions: COUNT, SUM, MAX, MIN, and AVG (NULL values discarded when aggregate functions are applied to a particular column)
3) Functions can be used in the SELECT clause or in a HAVING clause

4) Partition relation into subsets of tuples
   - Based on grouping attribute(s)
   - Apply function to each such group independently

5) GROUP BY clause: Specifies grouping attributes

6) If NULLs exist in grouping attribute then separate group created for all tuples with a NULL value in grouping attribute

7) HAVING clause provides a condition on the summary information

8) SELECT clause includes only the grouping attribute and the aggregate functions to be applied on each group of tuples.

9) WHERE clause limit the *tuples* to which functions are applied, the HAVING clause serves to choose *whole groups*

Question Bank:

Q1) Database table by name Loan_Records is given below.

Borrower	Bank_Manager	Loan_Amount
Ramesh	Sunderajan	10000.00
Suresh	Ramgopal	5000.00
Mahesh	Sunderajan	7000.00

What is the output of the following SQL query?

```
SELECT Count(*)
FROM  ( ( SELECT Borrower, Bank_Manager
        FROM Loan_Records) AS S
        NATURAL JOIN ( SELECT Bank_Manager, Loan_Amount
                FROM Loan_Records) AS T );
```

1) 3
2) 9
3) 5
4) 6

Q2) Which product is returned in a join query have no join condition:
1) Equijoins
2) Cartesian
3) Both Equijoins and Cartesian
4) None of the mentioned

Q3) Which join refers to join records from the write table that have no matching key in the left table are include in the result set:
1) Left outer join
2) Right outer join
3) Full outer join
4) Half outer join

Q4) Which operation are allowed in a join view:
1) UPDATE
2) INSERT
3) DELETE
4) All of the mentioned

Q5) Which view that contains more than one table in the top-level FROM clause of the SELECT statement:
1) Join view
2) Datable join view
3) Updatable join view
4) All of the mentioned

Q6) The following SQL is which type of join:

SELECT CUSTOMER_T. CUSTOMER_ID, ORDER_T. CUSTOMER_ID, NAME, ORDER_ID

FROM CUSTOMER_T,ORDER_T

WHERE CUSTOMER_T. CUSTOMER_ID = ORDER_T. CUSTOMER_ID ?
1) Equi-join
2) Natural join
3) Outer join
4) Cartesian join

Q7) The following SQL is which type of join:

SELECT CUSTOMER_T. CUSTOMER_ID, ORDER_T. CUSTOMER_ID, NAME, ORDER_ID

FROM CUSTOMER_T,ORDER_T ?
1) Equi-join
2) Natural join
3) Outer join
4) Cartesian join

Q8) Which of the following statements is true concerning subqueries?
1) Involves the use of an inner and outer query
2) Cannot return the same result as a query that is not a subquery

3) Does not start with the word SELECT
4) All of the mentioned

Q9) Suppose ORACLE relation R(A, B) currently has tuples {(1, 2), (1, 3), (3, 4)} and relation S(B, C) currently has {(2, 5), (4, 6), (7, 8)}. Consider the following two SQL queries SQ_1 and SQ_2 :

SQ_1 : Select * From R Full Join S On R.B = S.B;

SQ_2 : Select * From R Inner Join S On R.B = S.B;

The numbers of tuples in the result of the SQL query SQ_1 and the SQL query SQ_2 are given by:

1) 2 and 6 respectively
2) 6 and 2 respectively
3) 2 and 4 respectively
4) 4 and 2 respectively

Q10) Which of the following is true ? I. Implementation of self-join is possible in SQL with table alias. II. Outer-join operation is basic operation in relational algebra. III. Natural join and outer join operations are equivalent.

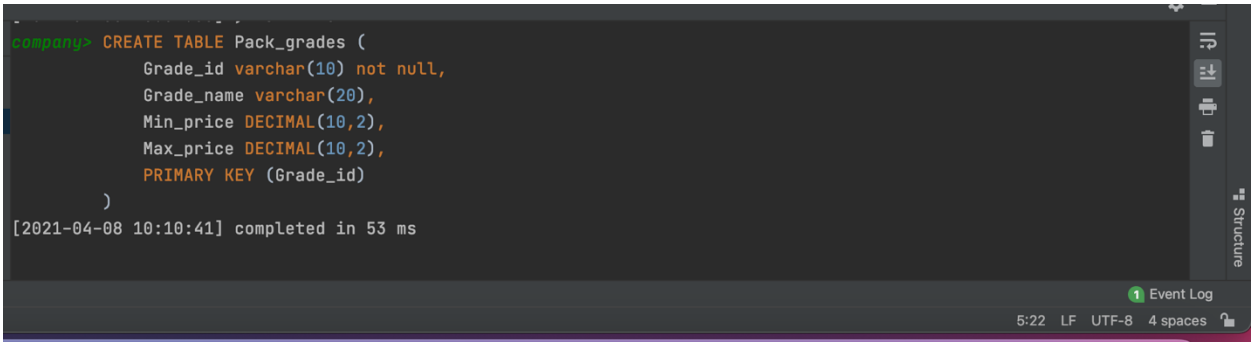
1) I and II are correct.
2) II and III are correct.
3) Only III is correct.
4) Only I is correct.

Student Work Area

Work:

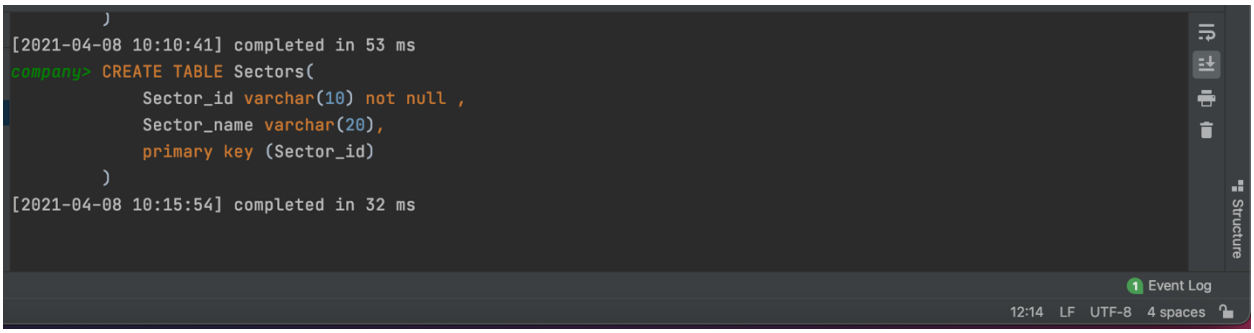
1. **Pack_grades Table**

```
CREATE TABLE Pack_grades (
   Grade_id varchar(10) not null,
   Grade_name varchar(20),
   Min_price DECIMAL(10,2),
   Max_price DECIMAL(10,2),
   PRIMARY KEY (Grade_id)
);
```



2. **Sectors Table**

```
CREATE TABLE Sectors(
   Sector_id varchar(10) not null ,
   Sector_name varchar(20),
   primary key (Sector_id)
);
```



3. **Packages Table**

CREATE TABLE Packages(

Pack_id varchar(10) not null ,

speed varchar(10),

Start_date date,

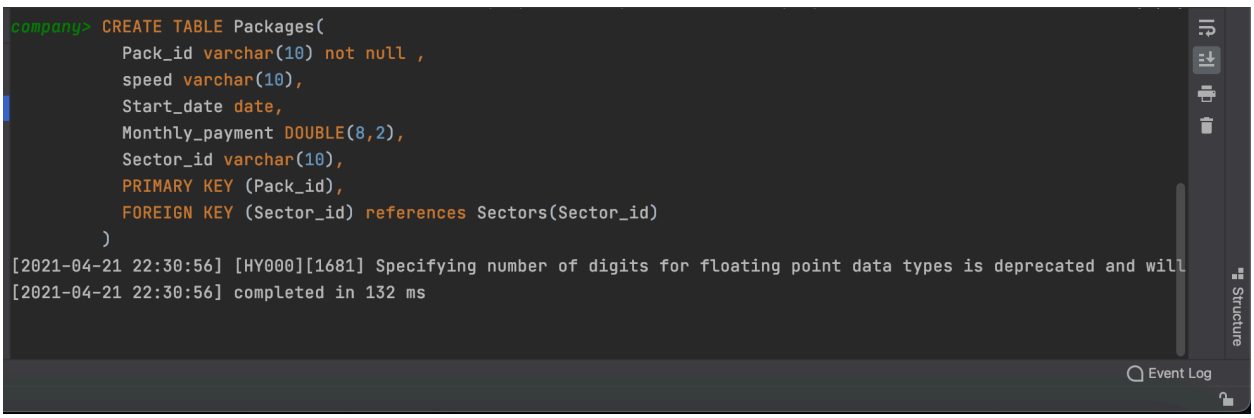
Monthly_payment DOUBLE(8,2),

Sector_id varchar(10),

PRIMARY KEY (Pack_id),

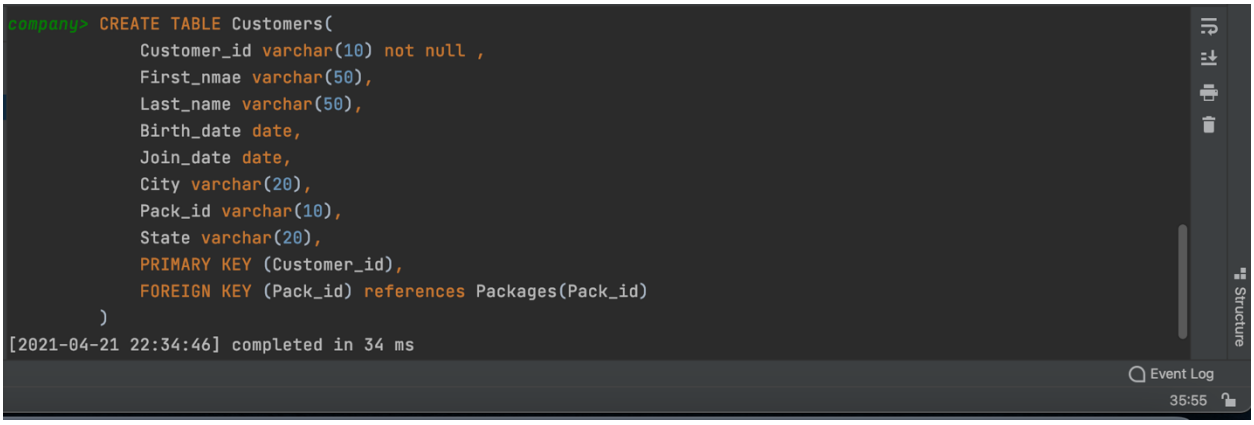
FOREIGN KEY (Sector_id) references Sectors(Sector_id)

);



4. Customers Table

```
CREATE TABLE Customers(
   Customer_id varchar(10) not null ,
   First_nmae varchar(50),
   Last_name varchar(50),
   Birth_date date,
   Join_date date,
   City varchar(20),
   Pack_id varchar(10),
   State varchar(20),
   PRIMARY KEY (Customer_id),
   FOREIGN KEY (Pack_id) references Packages(Pack_id)
);
```



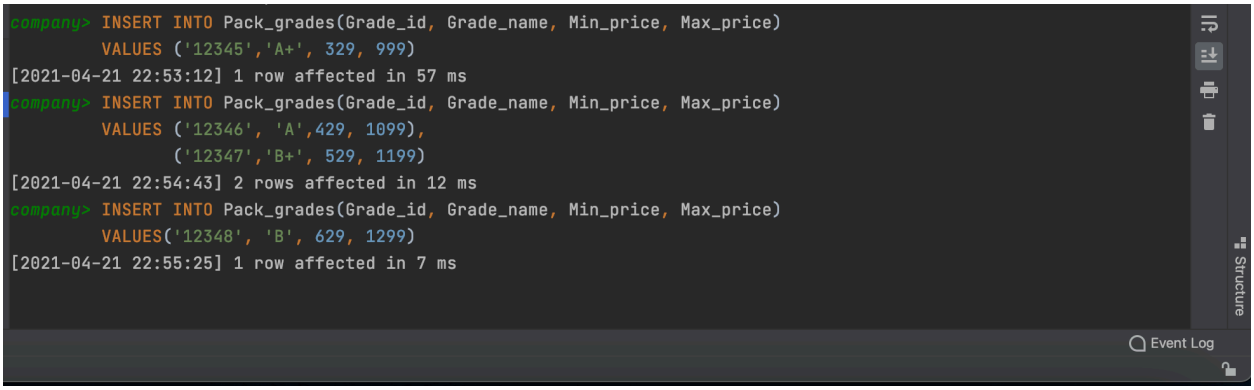
Now We have to Insert Some Data in the Database

1. Inserting Values in Pack_grades

```
INSERT INTO Pack_grades(Grade_id, Grade_name, Min_price, Max_price)
VALUES ('12345','A+', 329, 999);

INSERT INTO Pack_grades(Grade_id, Grade_name, Min_price, Max_price)
VALUES ('12346', 'A',429, 1099),
    ('12347','B+', 529, 1199);

INSERT INTO Pack_grades(Grade_id, Grade_name, Min_price, Max_price)
VALUES('12348', 'B', 629, 1299);
```

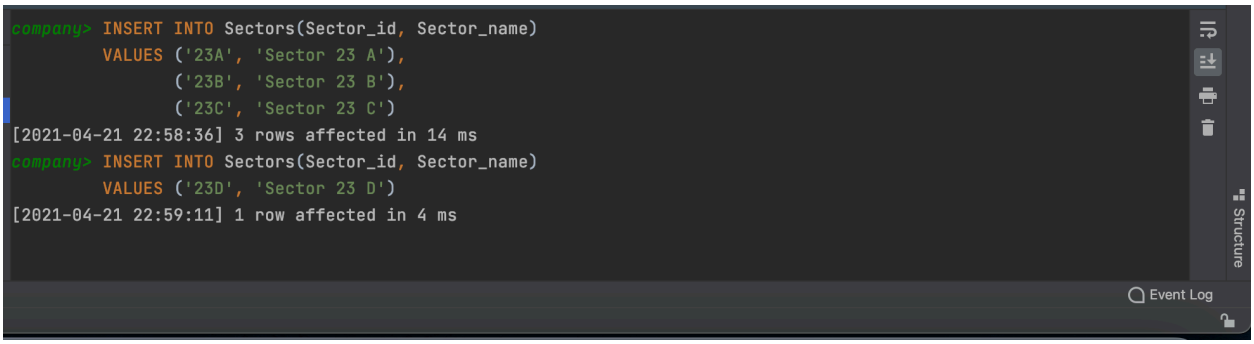


2. Inserting Values in Sectors

```
INSERT INTO Sectors(Sector_id, Sector_name)
VALUES ('23A', 'Sector 23 A'),
    ('23B', 'Sector 23 B'),
    ('23C', 'Sector 23 C');
```

INSERT INTO Sectors(Sector_id, Sector_name)

VALUES ('23D', 'Sector 23 D');



3. Inserting Values in Packages

INSERT INTO Packages(Pack_id, speed, Start_date, Monthly_payment, Sector_id)

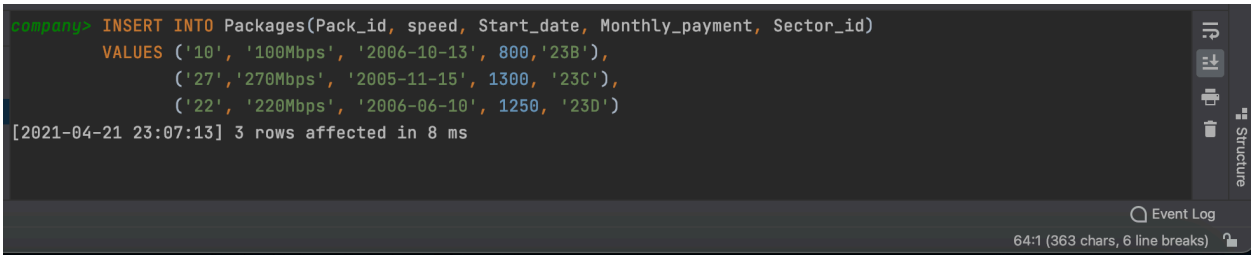
VALUES ('20', '200Mbps', '2020-08-12', 1000, '23A');

INSERT INTO Packages(Pack_id, speed, Start_date, Monthly_payment, Sector_id)

VALUES ('10', '100Mbps', '2006-10-13', 800,'23B'),

('27','270Mbps', '2005-11-15', 1300, '23C'),

('22', '220Mbps', '2006-06-10', 1250, '23D');



4. Inserting Values in Customers

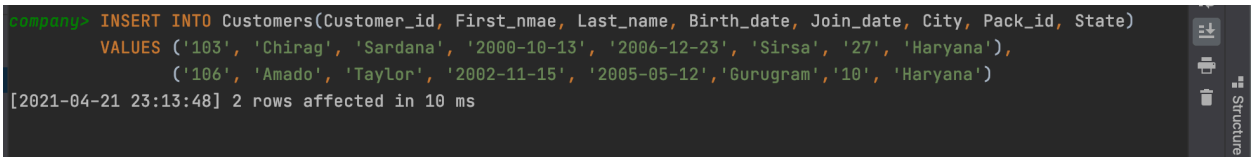
INSERT INTO Customers(Customer_id, First_nmae, Last_name, Birth_date, Join_date, City, Pack_id, State)

VALUES ('100', 'Ayush', 'Singh', '2001-05-10', '2006-05-10', 'Delhi','22', 'Delhi');

INSERT INTO Customers(Customer_id, First_nmae, Last_name, Birth_date, Join_date, City, Pack_id, State)

VALUES ('103', 'Chirag', 'Sardana', '2000-10-13', '2006-12-23', 'Sirsa', '27', 'Haryana'),

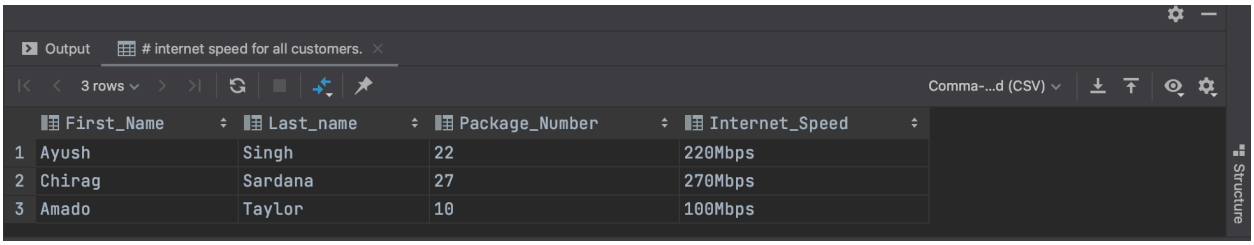
('106', 'Amado', 'Taylor', '2002-11-15', '2005-05-12','Gurugram','10', 'Haryana');



Algorithm/Flowchart/Code/Sample Outputs

1) Write a query to display first name, last name, package number and internet speed for all customers.

Command: SELECT First_name, Last_name, P.Pack_id As Package_Number, speed As Internet_Speed From Customers JOIN Packages P on P.Pack_id = Customers.Pack_id;



	First_Name	Last_name	Package_Number	Internet_Speed
1	Ayush	Singh	22	220Mbps
2	Chirag	Sardana	27	270Mbps
3	Amado	Taylor	10	100Mbps

2) Write a query to display first name, last name, package number and internet speed for all customers whose package number equals 22 or 27. Order the query in ascending order by last name.

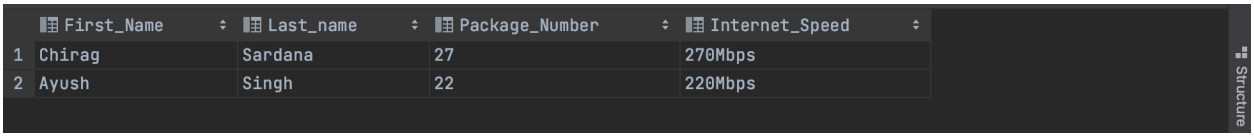
Command: SELECT First_name, Last_name, P.Pack_id As Package_Number,

speed As Internet_Speed

From Customers JOIN Packages P on P.Pack_id = Customers.Pack_id

WHERE P.Pack_id = '22' OR P.Pack_id = '27'

ORDER BY Last_name;

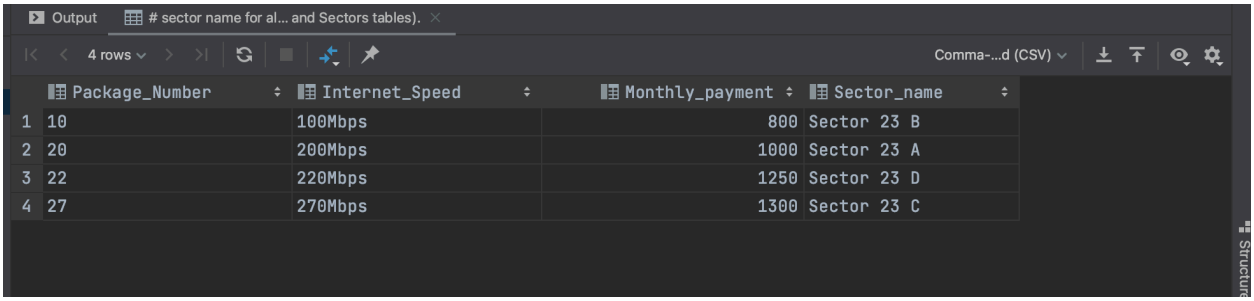


	First_Name	Last_name	Package_Number	Internet_Speed
1	Chirag	Sardana	27	270Mbps
2	Ayush	Singh	22	220Mbps

3) Display the package number, internet speed, monthly payment and sector name for all packages (*Packages* and *Sectors* tables).

Command: SELECT Pack_id As Package_Number, speed As Internet_Speed, Monthly_payment, Sectors.Sector_name

FROM Packages NATURAL JOIN Sectors;



	Package_Number	Internet_Speed	Monthly_payment	Sector_name
1	10	100Mbps	800	Sector 23 B
2	20	200Mbps	1000	Sector 23 A
3	22	220Mbps	1250	Sector 23 D
4	27	270Mbps	1300	Sector 23 C

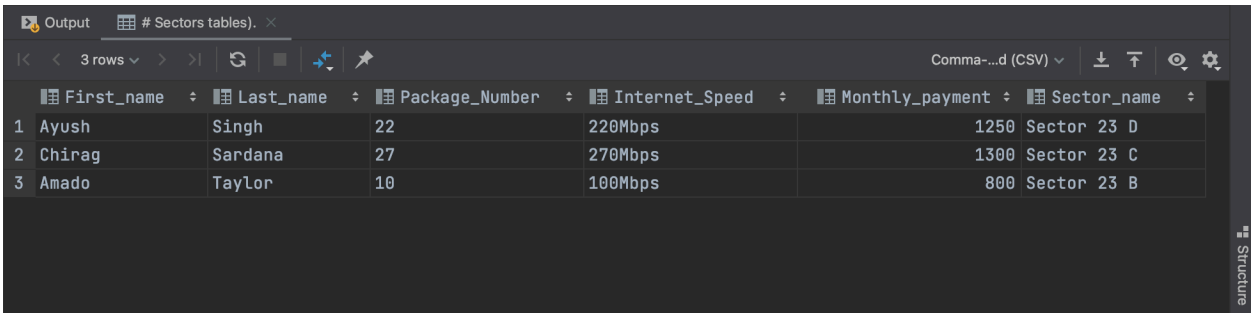
4) Display the customer name, package number, internet speed, monthly payment and sector name for all customers (*Customers*, *Packages* and *Sectors* tables).

Command: SELECT First_name, Last_name, P.Pack_id As Package_Number,

speed AS Internet_Speed, Monthly_payment, S.Sector_name

FROM Customers JOIN Packages P on P.Pack_id = Customers.Pack_id

JOIN Sectors S on S.Sector_id = P.Sector_id;



	First_name	Last_name	Package_Number	Internet_Speed	Monthly_payment	Sector_name
1	Ayush	Singh	22	220Mbps	1250	Sector 23 D
2	Chirag	Sardana	27	270Mbps	1300	Sector 23 C
3	Amado	Taylor	10	100Mbps	800	Sector 23 B

5) Display the customer name, package number, internet speed, monthly payment and sector name for all customers in the business sector (*Customers*, *Packages* and *Sectors* tables).

Some Extra Insertion Done according to Given Question

Command: INSERT INTO Sectors(sector_id, sector_name)
VALUES ('Business', 'Business');

```
company> INSERT INTO Sectors(sector_id, sector_name)
         VALUES ('Business', 'Business')
[2021-04-22 00:20:01] 1 row affected in 6 ms
```

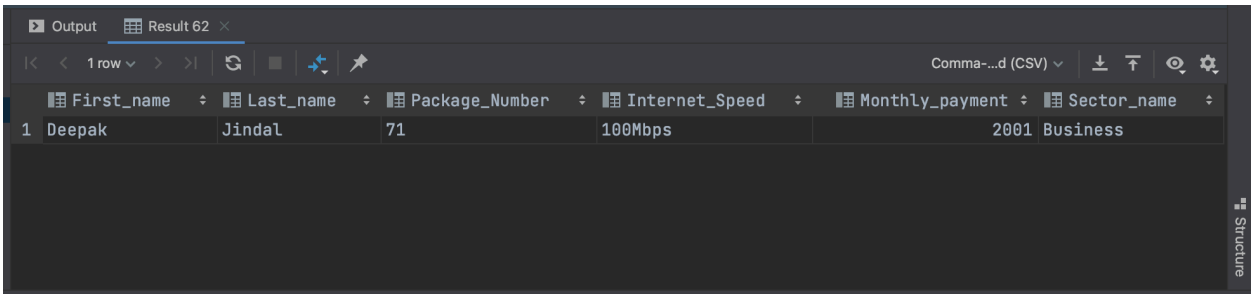
Command: INSERT INTO Packages(Pack_id, speed, Start_date, Monthly_payment, Sector_id)
VALUES ('71', '100Mbps', '2021-04-21',2001,'Business');

```
company> INSERT INTO Packages(Pack_id, speed, Start_date, Monthly_payment, Sector_id)
         VALUES ('71', '100Mbps', '2021-04-21',2001,'Business')
[2021-04-22 00:22:55] 1 row affected in 7 ms
```

Command: INSERT INTO Customers(Customer_id, First_nmae, Last_name, Birth_date, Join_date, City, Pack_id, State)
VALUES ('112','Deepak','Jindal','2001-08-26','2006-10-12','Delhi','71','Delhi');

```
company> INSERT INTO Customers(Customer_id, First_nmae, Last_name, Birth_date, Join_date, City, Pack_id, State)
         VALUES ('112','Deepak','Jindal','2001-08-26','2006-10-12','Delhi','71','Delhi')
[2021-04-22 00:25:30] 1 row affected in 33 ms
```

Command: SELECT First_name, Last_name, Packages.Pack_id As Package_Number,
Packages.speed As Internet_Speed, Packages.Monthly_payment, Sectors.Sector_name
FROM Customers NATURAL JOIN Packages NATURAL JOIN Sectors
WHERE Sector_name = 'Business';



6) Display the last name, first name, join date, package number, internet speed and sector name for all customers in the private sector who joined the company in the year 2006.

Some Extra Insertion Done according to Given Question

Command: INSERT INTO Sectors(sector_id, sector_name)

VALUES ('Private', 'Private');

Command: INSERT INTO Packages(Pack_id, speed, Start_date, Monthly_payment, Sector_id)

VALUES ('74', '100Mbps', '2021-04-21',2002,'Private');

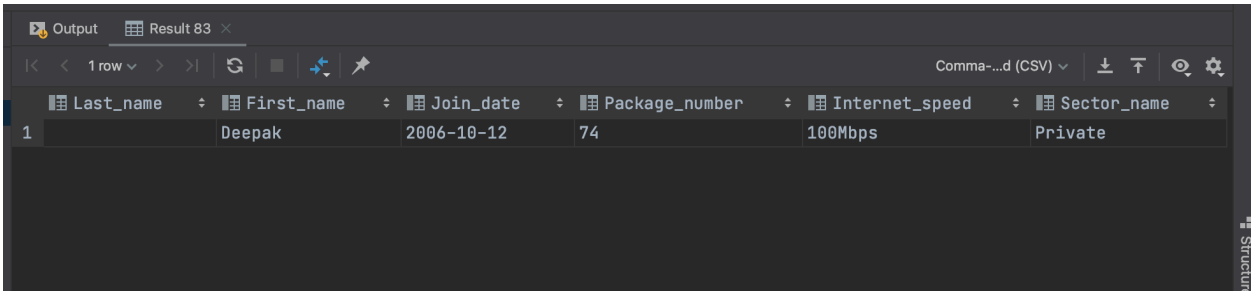
Command: INSERT INTO Customers(Customer_id, First_name, Last_name, Birth_date, Join_date, City, Pack_id, State)

VALUES ('115','Deepak',' ','2001-08-26','2006-10-12','Delhi','74','Delhi');

```
company> INSERT INTO Sectors(sector_id, sector_name)
         VALUES ('Private', 'Private')
[2021-04-22 00:28:06] 1 row affected in 7 ms
company> INSERT INTO Packages(Pack_id, speed, Start_date, Monthly_payment, Sector_id)
         VALUES ('74', '100Mbps', '2021-04-21',2002,'Business')
[2021-04-22 00:28:09] 1 row affected in 6 ms
company> INSERT INTO Customers(Customer_id, First_nmae, Last_name, Birth_date, Join_date, City, Pack_id, State)
         VALUES ('115','Deepak',' ','2001-08-26','2006-10-12','Delhi','74','Delhi')
[2021-04-22 00:28:12] 1 row affected in 5 ms
```

Command: SELECT Last_name, First_name, Join_date, P.Pack_id As Package_number,

speed As Internet_speed, Sector_name FROM Customers NATURAL JOIN Packages P NATURAL JOIN Sectors WHERE Sector_name = 'Private' AND YEAR(Join_date) = 2006;

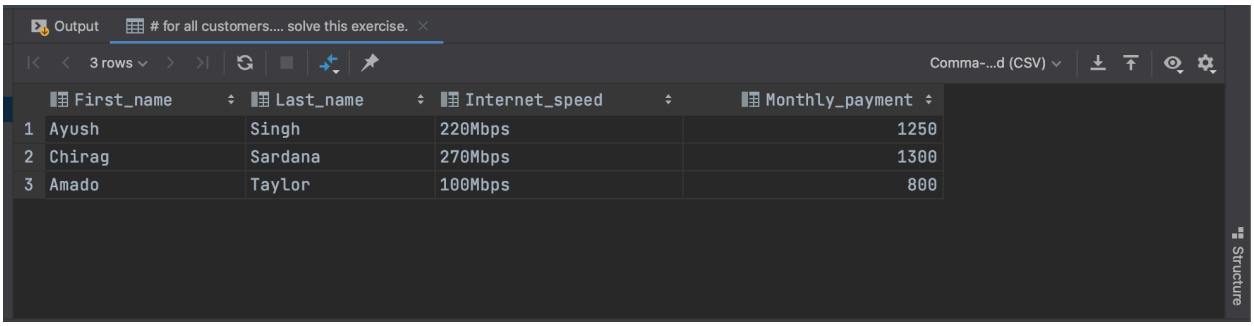


7) Display the package number, internet speed, monthly payment and package grade for all packages (*Packages* and *Pack_Grades* tables).

Command:

8) Display the first name, last name, internet speed and monthly payment for all customers. Use INNER JOIN to solve this exercise.

Command: SELECT First_name, Last_name, speed AS Internet_speed, Monthly_payment FROM Customers INNER JOIN Packages P on Customers.Pack_id = P.Pack_id;



9) Display the last name, first name and package number for all customers who have the same package number as customer named ‘Amado Taylor’ (*Customers* table).

Some Extra Insertion Done according to Given Question

Command: INSERT INTO Customers(Customer_id, First_nmae, Last_name, Birth_date, Join_date, City, Pack_id, State) VALUES ('109', 'Kanu', 'Sardana', '2000-10-13', '2006-10-12', 'Hansi','10','Haryana');

```
company> INSERT INTO Customers(Customer_id, First_nmae, Last_name, Birth_date, Join_date, City, Pack_id, State)
         VALUES ('109', 'Kanu', 'Sardana', '2000-10-13', '2006-10-12', 'Hansi','10','Haryama')
[2021-04-22 00:00:43] 1 row affected in 10 ms
```

Command: SELECT Last_name, First_name, P.Pack_id As Package_Number

FROM Customers JOIN Packages P on P.Pack_id = Customers.Pack_id

WHERE P.Pack_id In (SELECT Pack_id FROM Customers WHERE First_nmae ='Amado' AND Last_name = 'Taylor');

Output | Result 42

2 rows | Comma-...d (CSV)

	Last_name	First_name	Package_Number
1	Taylor	Amado	10
2	Sardana	Kanu	10

Structure | Event Log

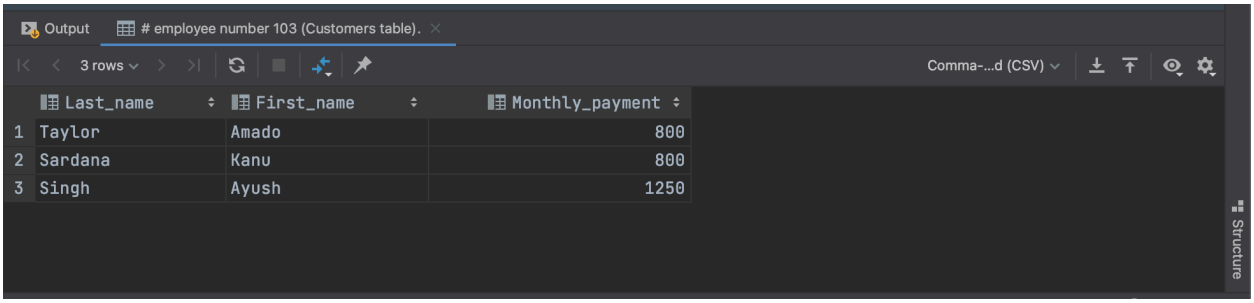
10) Display the last name, first name and monthly discount for all customers whose monthly discount is lower than the monthly discount of employee number 103 (*Customers* table).

Command: SELECT Last_name, First_nmae As First_name, Monthly_payment

FROM Customers NATURAL JOIN Packages

WHERE Monthly_payment < ALL(SELECT P2.Monthly_payment FROM Customers JOIN Packages P2

on P2.Pack_id = Customers.Pack_id WHERE Customer_id = '103');



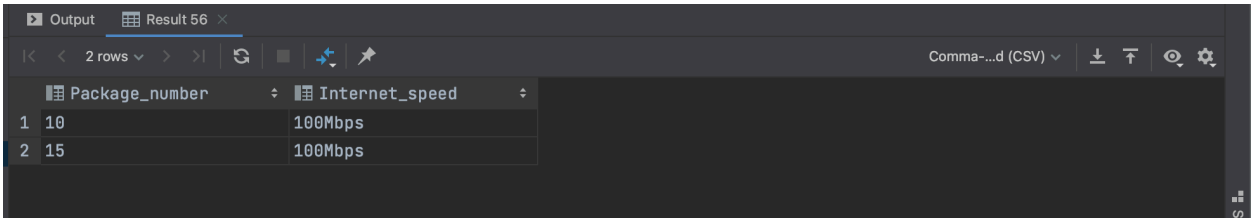
11) Display the package number and internet speed for all packages whose internet speed is equal to the internet speed of package number 10 (*Packages* table).

Some Extra Insertion Done according to Given Question

Command: INSERT INTO Packages(pack_id, speed, start_date, monthly_payment, sector_id) VALUES ('15','100Mbps', '2021-04-22', '2000','23A');

Command: SELECT Packages.Pack_id As Package_number, Packages.speed AS Internet_speed

FROM Packages JOIN Packages P on P.Pack_id = '10' AND Packages.speed = P.speed;



EXPERIMENT NO. 6

Student Name and Roll Number: Chirag Sardana and 19CSU071
Semester /Section: 4th / FS A1
Link to Code: https://github.com/chiragsardana/Question-In-Java/tree/master/DBMS/Experiment%206
Date:
Faculty Signature:
Marks:

Ojective(s):

To understand Aggregate functions and Group by Clause using SQL queries on the COMPANY database.

Outcome:

The students will understand the need of applying an aggregate function in order to get a single value from a set of values, thus, expressing the significance of the data it is computed from.

Problem Statement:

Consider the COMPANY database in Experiment 1 and execute the following queries:

1) Find the sum of the salaries of all employees, the maximum salary, the minimum salary, and the average salary.
2) Find the sum of the salaries of all employees of the 'Research'department, as well as the maximum salary, the minimum salary, and the average salary in this department.
3) Retrieve the total number of employees in the company.
4) Retrieve the number of employees in the 'Research' department.
5) Count the number of distinct salary values in the database
6) For each department, retrieve the department number, the number of employees in the department, and their average salary.
7) For each project, retrieve the project number, the project name, and the number of employees who work on that project.

8) For each project *on which more than two employees work,* retrieve the project number, the project name, and the number of employees who work on the project
9) For each project, retrieve the project number, the project name, and the number of employees from department 5 who work on the project.
10) For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than $40,000.

Background Study:

Aggregate Functions : Used to summarize information from multiple tuples into a single-tuple summary

1) Grouping : Create subgroups of tuples before summarizing
2) Built-in aggregate functions: COUNT, SUM, MAX, MIN, and AVG (NULL values discarded when aggregate functions are applied to a particular column)
3) Functions can be used in the SELECT clause or in a HAVING clause
4) Partition relation into subsets of tuples
   - Based on grouping attribute(s)
   - Apply function to each such group independently
5) GROUP BY clause: Specifies grouping attributes
6) If NULLs exist in grouping attribute then separate group created for all tuples with a NULL value in grouping attribute
7) HAVING clause provides a condition on the summary information
8) SELECT clause includes only the grouping attribute and the aggregate functions to be applied on each group of tuples.
9) WHERE clause limit the *tuples* to which functions are applied, the HAVING clause serves to choose *whole groups*

Question Bank:

Q1) Which of the following statements are TRUE about an SQL query? P : An SQL query can contain a HAVING clause even if it does not have a GROUP BY clause Q : An SQL query can contain a HAVING clause only if it has a GROUP BY clause R : All attributes used in the GROUP

BY clause must appear in the SELECT clause S : Not all attributes used in the GROUP BY clause need to appear in the SELECT clause

1) P and R
2) P and S
3) Q and R
4) Q and S

Q2) Consider a database table T containing two columns X and Y each of type integer. After the creation of the table, one record (X=1, Y=1) is inserted in the table. Let MX and My denote the respective maximum values of X and Y among all records in the table at any point in time. Using MX and MY, new records are inserted in the table 128 times with X and Y values being MX+1, 2*MY+1 respectively. It may be noted that each time after the insertion, values of MX and MY change. What will be the output of the following SQL query after the steps mentioned above are carried out?

SELECT Y FROM T WHERE X=7;

1) 127
2) 255
3) 129
4) 257

Q3) The employee information in a company is stored in the relation

Employee (name, sex, salary, deptName)

Consider the following SQL query
select deptName

from Employee

where sex = 'M'

group by deptName

having avg (salary) > (select avg (salary) from Employee)

It returns the names of the department in which

1) the average salary is more than the average salary in the company
2) the average salary of male employees is more than the average salary of all male employees in the company
3) the average salary of male employees is more than the average salary of employees in the same department
4) the average salary of male employees is more than the average salary in the company

Q4) Which of the following is aggregate function in SQL?

1) Avg
2) Select
3) Ordered by
4) distinct

Q5) Observe the given SQL query and choose the correct option.

```
SELECT branch_name, COUNT (DISTINCT customer_name)
FROM depositor, account
WHERE depositor.account_number = account.account_number
GROUP BY branch_id
```

1) The query is syntactically correct but gives the wrong answer
2) The query is syntactically wrong
3) The query is syntactically correct and gives the correct answer
4) The query contains one or more wrongly named clauses.

Q6) Choose the correct option regarding the query

```
SELECT branch_name, COUNT (DISTINCT customer_name)
FROM depositor, account
WHERE depositor.account_number = account.account_number
    GROUP BY branch_id
    HAVING avg(balance) = 10000;
```

1) The having clause checks whether the query result is true or not
2) The having clause does not check for any condition
3) The having clause allows only those tuples that have average balance 10000
4) None of the mentioned

Q7) Which keyword is used to rename the resulting attribute after the application of the aggregation function?

1) rename
2) as
3) replace
4) to

Q8) Consider the following ORACLE relations :

R (A, B, C) = {<1, 2, 3>, <1, 2, 0>, <1, 3, 1>, <6, 2, 3>, <1, 4, 2>, <3, 1, 4> } S (B, C, D) = {<2, 3, 7>, <1, 4, 5>, <1, 2, 3>, <2, 3, 4>, <3, 1, 4>}.

Consider the following two SQL queries SQ_1 and SQ_2 :

SQ_1 : SELECT R·B, AVG (S·B) FROM R, S WHERE R·A = S·C AND S·D < 7 GROUP BY R·B;
SQ_2 : SELECT DISTINCT S·B, MIN (S·C) FROM S GROUP BY S·B HAVING COUNT (DISTINCT S·D) > 1;

If M is the number of tuples returned by SQ_1 and N is the number of tuples returned by SQ_2 then

1) M = 4, N = 2
2) M = 5, N = 3
3) M = 2, N = 2
4) M = 3, N = 3

Q9) What values does the count(*) function ignore?
1) Repetitive values
2) Null values
3) Characters
4) Integers

Q10) State true or false: Any attribute which is present in the having clause without being aggregated must not be present in the group by clause.
1) True
2) False

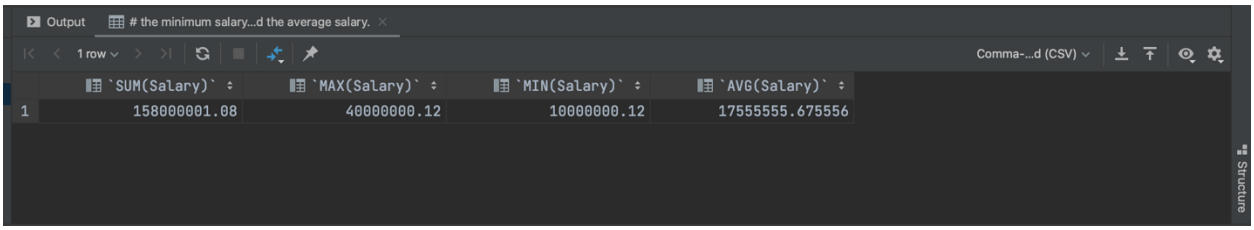
Student Work Area

Algorithm/Flowchart/Code/Sample Outputs

From the COMPANY database in Experiment 1

1) Find the sum of the salaries of all employees, the maximum salary, the minimum salary, and the average salary.

Command: SELECT SUM(Salary), MAX(Salary), MIN(Salary), AVG(Salary) FROM EMPLOYEE;

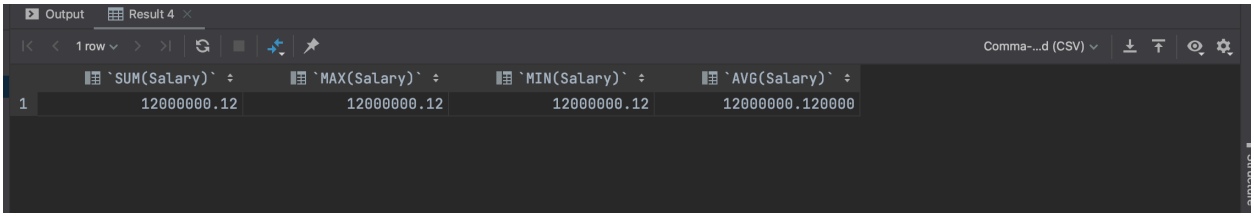


2) Find the sum of the salaries of all employees of the 'Research'department, as well as the maximum salary, the minimum salary, and the average salary in this department.

Command: SELECT SUM(Salary) , MAX(Salary), MIN(Salary), AVG(Salary)

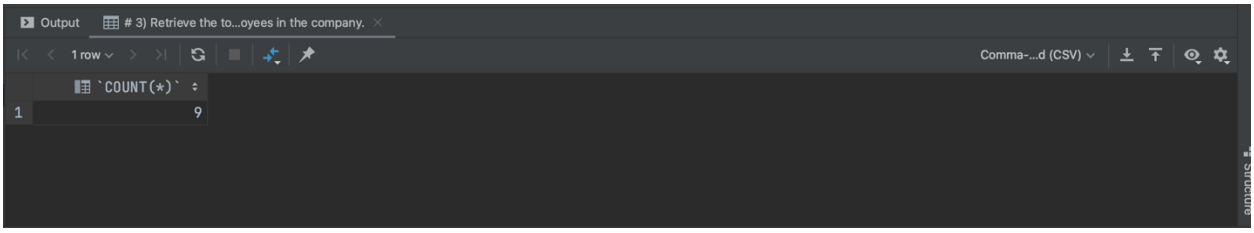
FROM EMPLOYEE JOIN DEPARTMENT D on D.Dnumber = EMPLOYEE.Dno

WHERE Dname = 'Research';



3) Retrieve the total number of employees in the company.

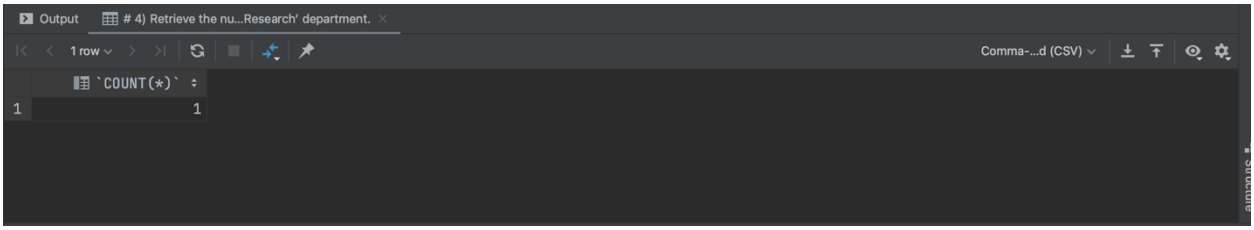
Command: SELECT COUNT(*) FROM EMPLOYEE;



4) Retrieve the number of employees in the ‘Research’ department.

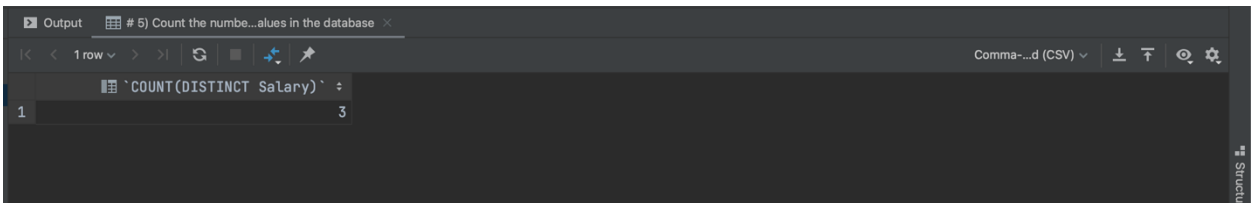
Command: SELECT COUNT(*) FROM EMPLOYEE JOIN DEPARTMENT D on D.Dnumber = EMPLOYEE.Dno

WHERE Dname = 'Research';



5) Count the number of distinct salary values in the database

Command: SELECT COUNT(DISTINCT Salary) FROM EMPLOYEE;

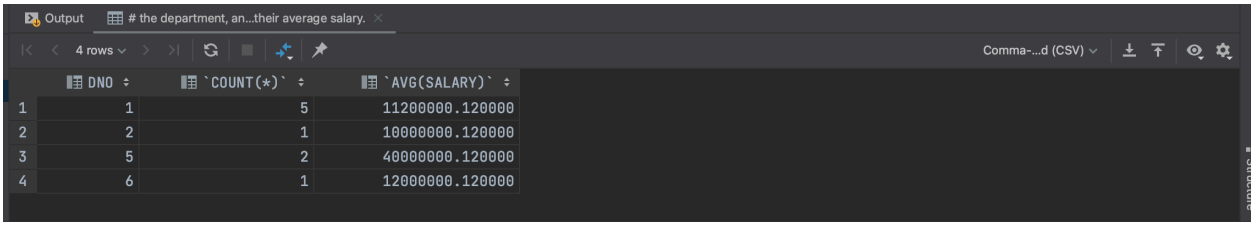


6) For each department, retrieve the department number, the number of employees in the department, and their average salary.

Command: SELECT DNO, COUNT(*), AVG(SALARY)

FROM EMPLOYEE

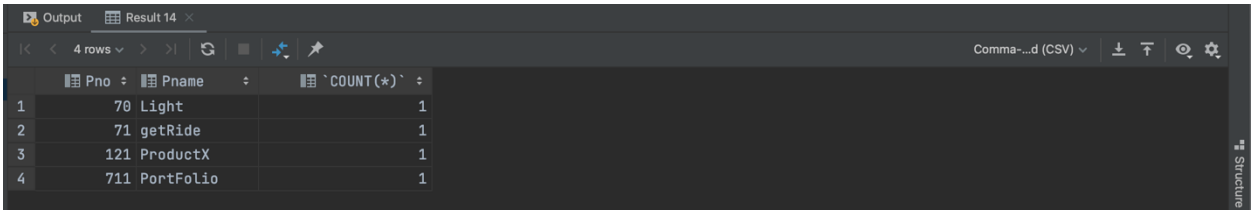
GROUP BY DNO;



7) For each project, retrieve the project number, the project name, and the number of employees who work on that project.

Command: SELECT Pno, Pname, COUNT(*) FROM WORKS_ON JOIN PROJECT P on P.Pnumber = WORKS_ON.Pno

GROUP BY Pno, Pname;



8) For each project *on which more than two employees work,* retrieve the project number, the project name, and the number of employees who work on the project

Some Extra Insertion Done according to Given Question

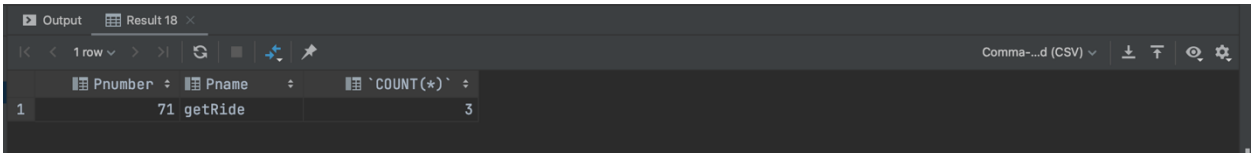
Command: INSERT INTO WORKS_ON(Essn, Pno, Hours)

VALUES ('123456785',71,12.3),

('123456786','71',14);

Command: SELECT Pnumber, Pname, COUNT(*) FROM PROJECT JOIN WORKS_ON WO on PROJECT.Pnumber = WO.Pno

GROUP BY Pnumber, Pname Having COUNT(*) > 2;



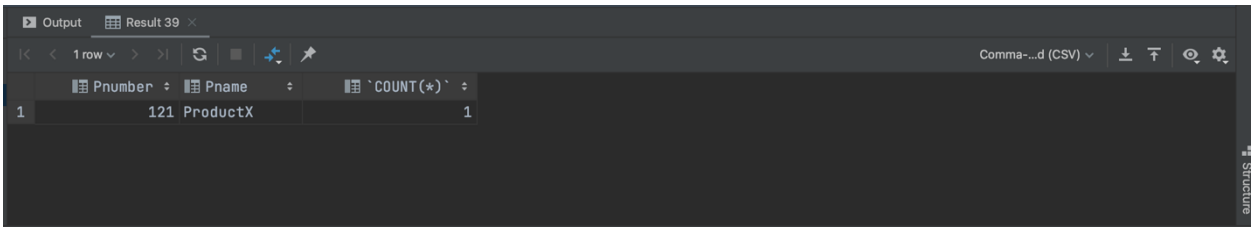
9) For each project, retrieve the project number, the project name, and the number of employees from department 5 who work on the project.

Command: SELECT Pnumber, Pname, COUNT(*)

FROM PROJECT JOIN WORKS_ON WO on PROJECT.Pnumber = WO.Pno

Join EMPLOYEE E on E.Ssn = WO.Essn AND Dno = 5

GROUP BY Pnumber, Pname;



10) For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than $40,000.

Some Extra Insertion Done according to Given Question

Command: INSERT INTO EMPLOYEE(Fname, Minit, Lname, Ssn, Bdate, Address, Sex, Salary, Dno)

VALUES ('Deepanshu',' ','Goel','123456532','2002-10-12','Delhi','M',50000, 1);

Command: SELECT DNUMBER, COUNT(*)

FROM DEPARTMENT, EMPLOYEE

WHERE DNUMBER = DNO AND SALARY > 40000 AND

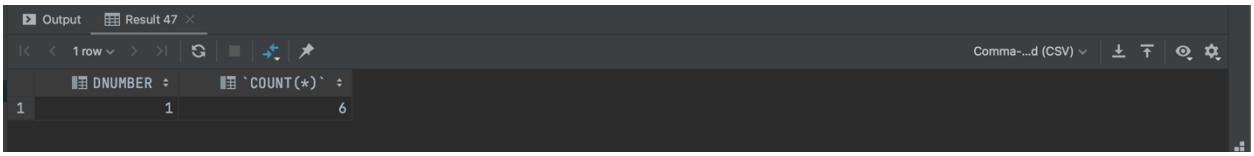
DNO IN (SELECT DNO

FROM EMPLOYEE

GROUP BY DNO

HAVING COUNT(*) > 5)

GROUP BY DNUMBER;


Output
Result 47
1 row
Comma-...d (CSV)
DNUMBER
`COUNT(*)`
1
1
6

EXPERIMENT NO. 7

Student Name and Roll Number: Chirag Sardana and 19CSU071
Semester /Section: 4th / FS A1
Link to Code: https://github.com/chiragsardana/Question-In-Java/tree/master/DBMS/Experiment%207
Date:
Faculty Signature:
Marks:

Objective(s):

To familiarize with nested SQL queries on the COMPANY database.

Outcome:

The students will understand how to devise independent and correlated nested queries.

Problem Statement:

Consider the COMPANY database in Experiment 1 and execute the following queries:

1) Make a list of all project numbers for projects that involve an employee whose last name is 'Smith', either as a worker or as a manager of the department that controls the project.
2) Select the Essns of all employees who work on the same project and hours as some project that employee 'John Smith' (whose Ssn = '123456789') works on.
3) Return the names of employees whose salary is greater than the salary of all the employees in department 5
4) Retrieve the name of each employee who has a dependent with the same first name and is the same sex as the employee
5) Retrieve the names of employees who have no dependents
6) List the names of managers who have at least one dependent using EXISTS and NOT EXISTS functions
7) Retrieve the name of each employee who works on all the projects controlled by department number 5 using EXISTS and NOT EXISTS functions
8) Retrieve the names of all employees who have two or more dependents
9) Retrieves the names of all employees who work on only one project

Background Study:

1) SQL allows queries that check whether an attribute value is NULL: IS or IS NOT NULL

2) Nested Queries: Complete select-from-where blocks within WHERE clause of another query
   - Nested Queries generally return a table (relation)

3) Comparison Operator IN:
   - Compares value *v* with a set (or multiset) of values *V*
   - Evaluates to TRUE if *v* is one of the elements in *V*

4) = ANY (or = SOME) Operator: Returns TRUE if the value *v* is equal to some value in the set *V* (equivalent to IN)

5) ALL Operator: (*v* > ALL *V*) returns TRUE if the value *v* is greater than *all* the values in the set (or multiset) *V*.
   - Other operators that can be combined with ANY (or SOME) and ALL: >, >=, <, <=, and <>

6) Possible ambiguity among attribute names if attributes of the same name exist—one in a relation in the FROM clause of the outer query, and another in a relation in the FROM clause of the nested query.
   - Thumb Rule: reference to an unqualified attribute refers to the relation declared in the innermost nested query.
7) Whenever a condition in the WHERE clause of a nested query references some attribute of a relation declared in the outer query, the two queries are said to be correlated.
   - In a correlated query, the nested query is evaluated once for each tuple (or combination of tuples) in the outer query

8) EXISTS function: Check whether the result of a correlated nested query is empty or not.

9) EXISTS and NOT EXISTS are typically used in conjunction with a correlated nested query.

10) EXISTS(Q): returns TRUE if there is at least one tuple in the result of the nested query Q, and it returns FALSE otherwise.

11) NOT EXISTS(Q): returns TRUE if there are no tuples in the result of nested query Q, and it returns FALSE otherwise.

Question Bank:

Q1) Database table by name Loan_Records is given below.

Borrower	Bank_Manager	Loan_Amount
Ramesh	Sunderajan	10000.00
Suresh	Ramgopal	5000.00
Mahesh	Sunderajan	7000.00

What is the output of the following SQL query?

```
SELECT Count(*)
FROM  ( ( SELECT Borrower, Bank_Manager
        FROM Loan_Records) AS S
        NATURAL JOIN ( SELECT Bank_Manager, Loan_Amount
                FROM Loan_Records) AS T );
```

1) 3
2) 9
3) 5
4) 6

Q2) A relational schema for a train reservation database is given below. Passenger (pid, pname, age) Reservation (pid, class, tid)

Table: Passenger

pid	pname	age
0	Sachin	65
1	Rahul	66
2	Sourav	67
3	Anil	69

Table : Reservation

pid	class	tid
0	AC	8200
1	AC	8201
2	SC	8201
5	AC	8203
1	SC	8204
3	AC	8202

What pids are returned by the following SQL query for the above instance of the tables?

SELECT pid

FROM Reservation ,

WHERE class 'AC' AND

*EXISTS (SELECT **

FROM Passenger

WHERE age > 65 AND

Passenger. pid = Reservation.pid)

1) 1,0
2) 1,2
3) 1,3
4) 1,5

Q3) Consider the following relational schema:

Suppliers(sid:integer, sname:string, city:string, street:string)
Parts(pid:integer, pname:string, color:string)
Catalog(sid:integer, pid:integer, cost:real)

Consider the following relational query on the above database:
SELECT S.sname

FROM Suppliers S

WHERE S.sid NOT IN (SELECT C.sid

FROM Catalog C

WHERE C.pid NOT IN (SELECT P.pid

FROM Parts P

WHERE P.color<> 'blue'))

Assume that relations corresponding to the above schema are not empty. Which one of the following is the correct interpretation of the above query?

1) Find the names of all suppliers who have supplied a non-blue part.
2) Find the names of all suppliers who have not supplied a non-blue part.
3) Find the names of all suppliers who have supplied only blue parts.
4) Find the names of all suppliers who have not supplied only blue parts.
5) None

Q4) Consider the table employee(empId, name, department, salary) and the two queries Q1 ,Q2 below. Assuming that department 5 has more than one employee, and we want to find the employees who get higher salary than anyone in the department 5, which one of the statements is TRUE for any arbitrary employee table?

Q1 : *Select e.empId*

From employee e

Where not exists

*(Select * From employee s where s.department = "5" and*

s.salary >=e.salary)

Q2 : *Select e.empId*

From employee e

Where e.salary > Any

(Select distinct salary From employee s Where s.department = "5")

1) Q1 is the correct query
2) Q2 is the correct query
3) Both Q1 and Q2 produce the same answer.
4) Neither Q1 nor Q2 is the correct query

Q5) Consider the following relational schema:

employee(empId, empName, empDept)
customer(custId, custName, salesRepId, rating)

salesRepId is a foreign key referring to empId of the employee relation. Assume that each employee makes a sale to at least one customer. What does the following query return?

```
SELECT empName
    FROM employee E
    WHERE NOT EXISTS ( SELECT custId
                FROM customer C
                WHERE C.salesRepId = E.empId
                 AND C.rating <> `GOOD`);
```

1) Names of all the employees with at least one of their customers having a 'GOOD' rating.

2) Names of all the employees with at most one of their customers having a 'GOOD' rating.

3) Names of all the employees with none of their customers having a 'GOOD' rating.

4) Names of all the employees with all their customers having a 'GOOD' rating.

Q6) The _____ clause allows us to select only those rows in the result relation of the ____ clause that satisfy a specified predicate.

1) Where, from
2) From, select
3) Select, from
4) From, where

Q7) The relation book (title, price) contains the titles and prices of different books. Assuming that no two books have the same price, what does the following SQL query list?

```
select title
from book as B
where (select count(*)
  from book as T
  where T.price > B.price) < 5
```

1) Titles of the four most expensive books
2) Title of the fifth most inexpensive book
3) Titles of the five most expensive books
4) Titles of the five most expensive books

Q8) Consider the following Employee table

ID	salary	DeptName
1	10000	EC
2	40000	EC
3	30000	CS
4	40000	ME
5	50000	ME

6 60000 ME

7 70000 CS

How many rows are there in the result of following query?

SELECT E.ID

FROM Employee E

WHERE EXISTS (SELECT E2.salary

FROM Employee E2

WHERE E2.DeptName = 'CS'

AND E.salary > E2.salary)

1) 0
2) 4
3) 5
4) 6

Q9) The _____ connective tests for set membership, where the set is a collection of values produced by a select clause. The _____ connective tests for the absence of set membership.

1) Or, in
2) Not in, in
3) In, not in
4) In, or

Q10) Consider the following relation: Cinema (theatre, address, capacity)

Which of the following options will be needed at the end of the SQL query

SELECT P1. address

FROM Cinema P1

Such that it always finds the addresses of theatres with maximum capacity?

1) WHERE P1. Capacity> = All (select P2. Capacity from Cinema P2)
2) WHERE P1. Capacity> = Any (select P2. Capacity from Cinema P2)
3) WHERE P1. Capacity > All (select max(P2. Capacity) from Cinema P2)
4) WHERE P1. Capacity > Any (select max (P2. Capacity) from Cinema P2)

Student Work Area

Algorithm/Flowchart/Code/Sample Outputs

From the COMPANY database in Experiment 1

1) Make a list of all project numbers for projects that involve an employee whose last name is ‘Smith’, either as a worker or as a manager of the department that controls the project.

Command: (SELECT DISTINCT PNUMBER

FROM PROJECT, WORKS_ON, EMPLOYEE

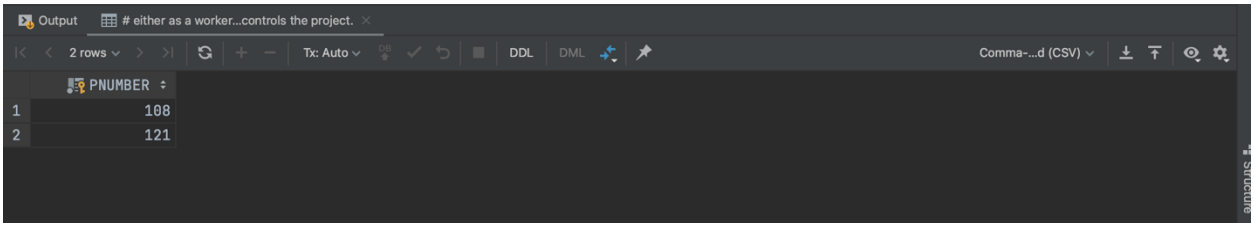
WHERE PNUMBER = PNO AND ESSN = SSN AND LNAME = 'Smith')

UNION

(SELECT DISTINCT PNUMBER

FROM PROJECT, DEPARTMENT, EMPLOYEE

WHERE DNUM = DNUMBER AND Mgr_ssn = SSN AND LNAME = 'Smith');



2) Select the Essns of all employees who work on the same project and hours as some project that employee ‘John Smith’ (whose Ssn = ‘123456789’) works on.

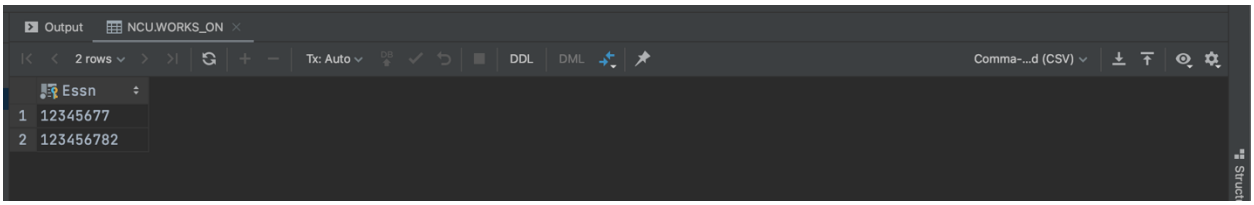
Some Extra Insertion Done according to Given Question

Command: INSERT INTO WORKS_ON(Essn, Pno, Hours)

VALUES ('12345677', 71, 12.1);

Command: SELECT Essn FROM WORKS_ON

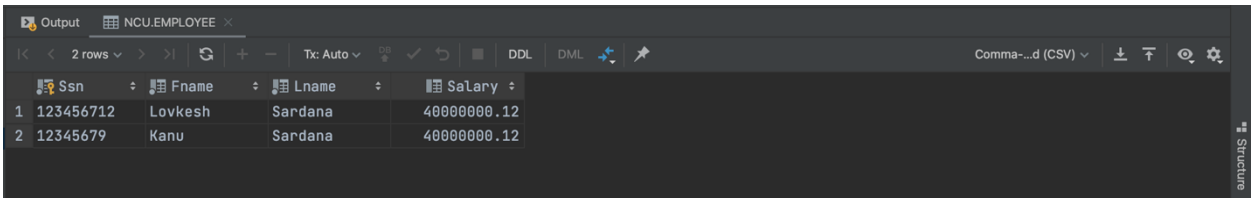
WHERE Hours IN (SELECT Hours FROM WORKS_ON JOIN EMPLOYEE E on E.Ssn = WORKS_ON.Essn AND E.Ssn ='12345677');



3) Return the names of employees whose salary is greater than the salary of all the employees in department 5

Command: SELECT Ssn, Fname, Lname, Salary FROM EMPLOYEE WHERE Salary >

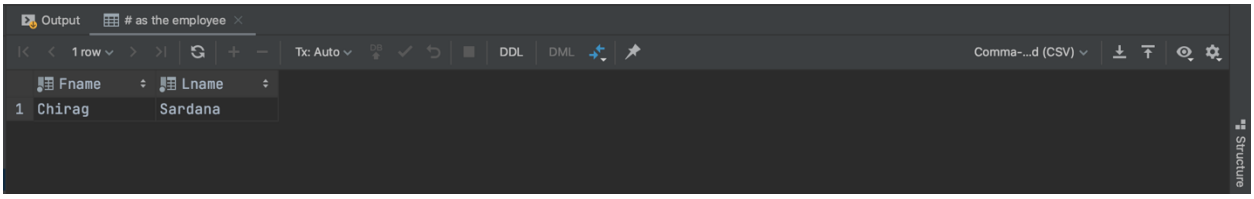
ALL (SELECT Salary FROM EMPLOYEE WHERE EMPLOYEE.Dno = 5);



4) Retrieve the name of each employee who has a dependent with the same first name and is the same sex as the employee

Command: Select Fname, Lname FROM EMPLOYEE JOIN DEPENDENT D on EMPLOYEE.Ssn = D.Essn AND EMPLOYEE.Fname = D.Dependent_name

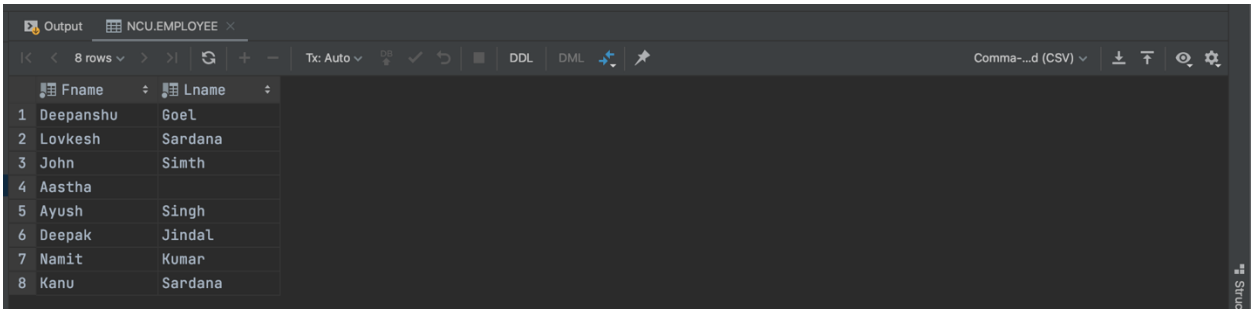
AND EMPLOYEE.Sex = D.Sex;



5) Retrieve the names of employees who have no dependents

Command: SELECT Fname, Lname FROM EMPLOYEE

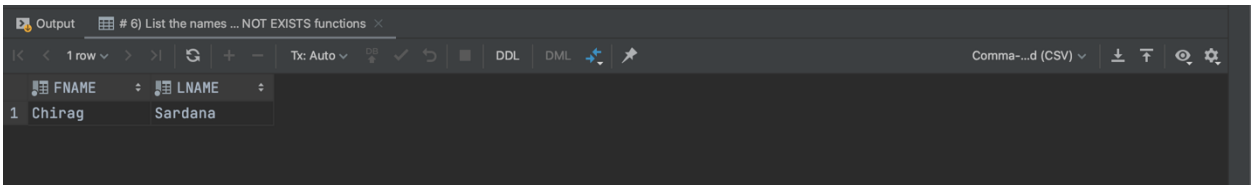
WHERE Ssn NOT IN (SELECT DISTINCT Ssn FROM EMPLOYEE JOIN DEPENDENT D on EMPLOYEE.Ssn = D.Essn);



6) List the names of managers who have at least one dependent using EXISTS and NOT EXISTS functions

Command: SELECT FNAME, LNAME

```
FROM EMPLOYEE
WHERE EXISTS (SELECT *
FROM DEPENDENT
WHERE SSN = ESSN)
AND
EXISTS (SELECT *
FROM DEPARTMENT
WHERE SSN = Mgr_ssn);
```



7) Retrieve the name of each employee who works on all the projects controlled by department number 5 using EXISTS and NOT EXISTS functions

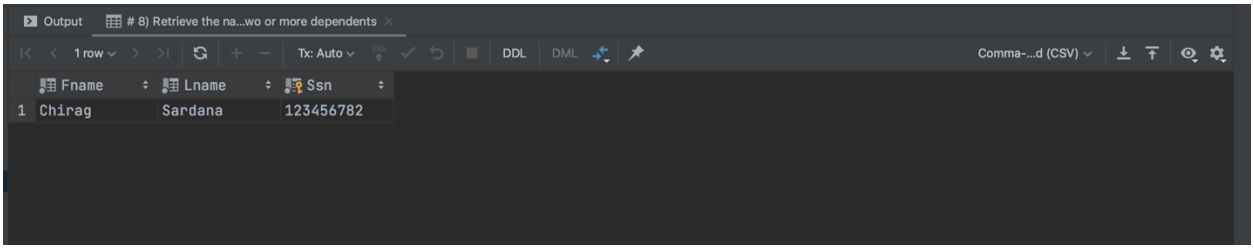
```
Command: SELECT FNAME, LNAME
FROM EMPLOYEE
WHERE ( (SELECT PNO
FROM WORKS_ON
WHERE SSN = ESSN)
IN
(SELECT PNUMBER
FROM PROJECT
WHERE DNUM = 5) );
```

8) Retrieve the names of all employees who have two or more dependents

Command: SELECT Fname, Lname, Ssn FROM EMPLOYEE

WHERE Ssn IN (SELECT Ssn FROM EMPLOYEE JOIN DEPENDENT D on EMPLOYEE.Ssn = D.Essn

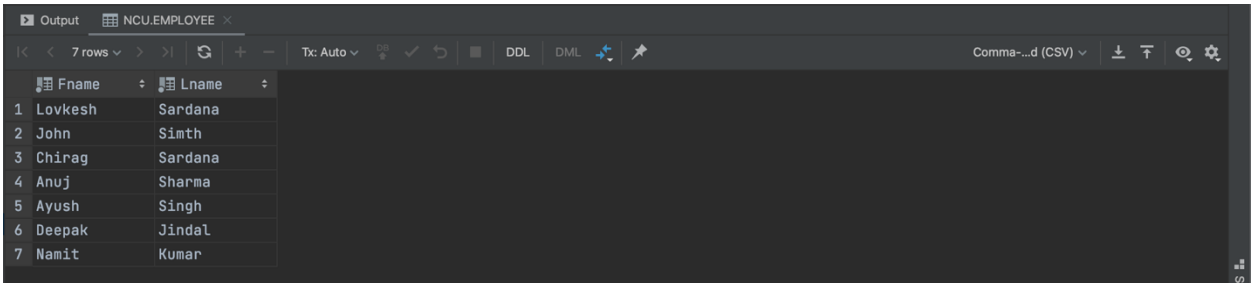
GROUP BY Ssn HAVING COUNT(*) >= 2);



9) Retrieves the names of all employees who work on only one project

Command: SELECT Fname, Lname FROM EMPLOYEE

WHERE Ssn IN (SELECT DISTINCT Essn FROM WORKS_ON GROUP BY Essn HAVING COUNT(*) =1);



EXPERIMENT NO. 8

Student Name and Roll Number:
Semester /Section:
Link to Code:
Date:
Faculty Signature:
Marks:

Objective(s):

Identifying contrast between Relational Databases and NoSQL, thereby recognizing their applications.

Outcome:

The students will install MongoDB shell and familiarize themselves with it.

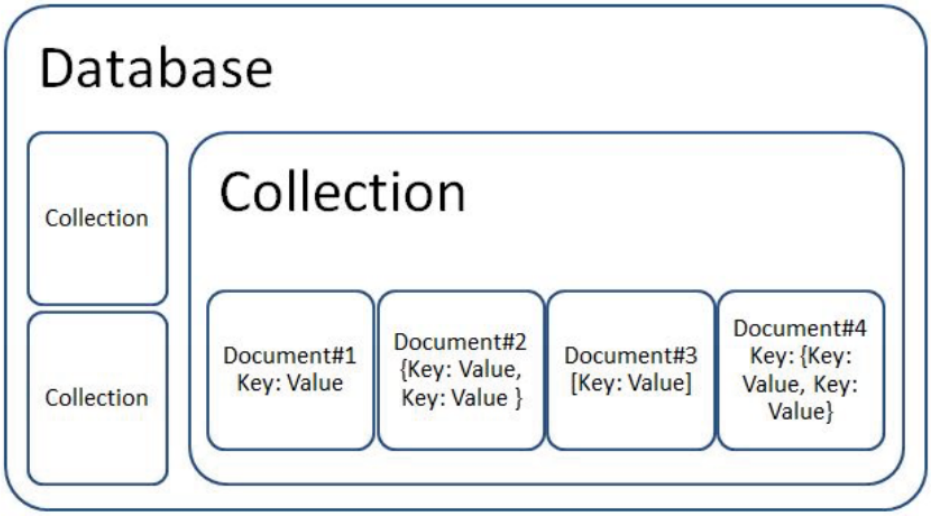
Problem Statement:

MongoDB installation and shell familiarity.

Background Study:

MongoDB is a Schema less database. A database in MongoDB contains collections and inside collections we have documents.

- Database is a physical container for collections. A single MongoDB server typically has multiple databases.
- Collection is a group of MongoDB documents.
- Documents within a collection can have different fields. A document is a set of key-value pairs and have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data.



RDBMS
• Relational database • Need to design your tables, data structure, relations first. • Supports SQL query language • Table based • Row based • Column based • Each row will have same number of columns • Primary Key • Contains schema which is predefined

MongoDB
• Non-relational and document-oriented database • You can start coding without worrying about tables and modify your objects at a lesser cost of development. • Supports JSON query language • Collection based and key-value pair • Document based • Field based • Each document can have different number of fields • Primary Key (Default key **_id** provided by MongoDB itself) • Contains dynamic schema

Question Bank:

Q1) _____ command displays the list of databases.

1) show db
2) show dbs
3) show data
4) display dbs

Q2) Which of the following operation is used to switch to new database mydb ?

1) use dbs
2) use db
3) use mydb
4) use mydbs

Q3) The command to check existence of collection is :

1) show collection
2) show collections
3) show collect
4) None of the mentioned

Q4) Point out the correct statement:

1) A database is a set of key-value pairs
2) A MongoDB deployment hosts a number of databases
3) A document holds a set of collections
4) All of the mentioned

Q5) MongoDB stores all documents in:

1) tables
2) collections
3) rows
4) all of the mentioned

Q6) BSON is a binary representation of _____ documents,

1) JSON
2) XML
3) JScript

4) All of the mentioned

Q7) MongoDB documents are composed of field-and-value pairs and have the following structure:

1) <field1:; value1>
2) <field1: value1;>
3) <field1: value1>
4) none of the mentioned

Q8) The _____ field is always the first field in the document.

1) _id
2) id
3) Ob_id
4) None of the mentioned

Q9) The maximum size of a MongoDB document is

1) 2 MB
2) 16 MB
3) 12 MB
4) There is no maximum size. It depends on RAM

Q10) Which of the following statements is true?

1) MongoDB cannot be used as a file system.
2) MongoDB can run over single servers only.
3) Embedded documents and arrays reduce need for joins.
4) None

Student Work Area

Algorithm/Flowchart/Code/Sample Outputs

EXPERIMENT NO. 9

Student Name and Roll Number:
Semester /Section:
Link to Code:
Date:
Faculty Signature:
Marks:

Objective(s):

Create COMPANY database using NoSQL database - MongoDB.

Outcome:

The students will be able to perform basic CRUD operations in MongoDB.

Problem Statement:

1) Create a COMPANY database with the following collections:
   - Employee with fields Emp ID, Ename, Age, Mobile, Email, Address, Dno
   - Department with fields Dnumber, Dname, Dlocation
   - Project with fields Pname, Pnumber, Plocation, ControlDept

2) Insert 5 documents in each collection using different versions of “Insert” command.

3) Select all documents in a collection.

4) Retrieve the details of all projects that are being run in department number 50.

5) Retrieve the details of the first project that is being run in department number 50 in formatted manner.

6) Return the formatted/structured project details of all projects in department number “5” and project name as “ProductZ”.

7) Return the project details (including project name, project number and department number and excluding project location and default primary key) of all projects with department number “5” or project name as “ProductZ”.

8) Return the formatted/structured project details of all projects with department number less than “10” and project name as “ProductZ” or project location as “Delhi”.

Background Study:

1) **use:** This command is used to create a database in MongoDB. It will return an existing database or will create a new database if it dosen’t exist.

2) **createCollection( ):** To create a collections in a database

3) MongoDB provides the following methods to insert documents into a collection:

4) **db.collection.insert():** used to insert one or multiple documents. To insert multiple documents in a single query, pass an array of documents in insert() command.

5) **db.collection.insertOne():** used to insert only one document.

6) **db.collection.insertMany():** used to insert multiple documents. To insert multiple documents in a single query, pass an array of documents in insert() command.

7) **db.collection.find():** this method is provided in MongoDB to read documents from a collection. It returns a **cursor** to the matching documents. db.collection.find(<query filter>, <projection>) accepts second optional parameter that is list of fields that you want to retrieve in the form { field1: <value>, field2: <value> ... }.

8) **db.collection.findOne()**: it returns the first occurrence of document, otherwise null.

Question Bank:

Q1) Which of the following method is used to query documents in collections ?

1) find
2) move
3) shell
4) replace

Q2) When you query a collection, MongoDB returns a _____ object that contains the results of the query.

1) row
2) cursor
3) colums
4) none of the mentioned

Q3) Which of the following method is called while accessing documents using the array index notation ?

1) cur.toArray()
2) cursor.toArray()
3) doc.toArray()
4) all of the mentioned

Q4) The mongo shell and the drivers provide several cursor methods that call on the cursor returned by the _____ method to modify its behavior.

1) cursor()
2) find()
3) findc()
4) none of the mentioned

Q5) Which of the following method corresponds to Order by clause in SQL ?

1) sort()
2) order()
3) orderby()
4) all of the mentioned

Q6) A query may include a _____ that specifies the fields from the matching documents to return.

1) selection
2) projection
3) union
4) none of the mentioned

Q7) In MongoDB, _____ operations modify the data of a single collection.

1) CRUD
2) GRID

3) READ
4) All of the mentioned

Q8) Which of the following operation adds a new document to the users collection ?

1) add
2) insert
3) truncate
4) drop

Q9) Which of the following is a valid MongoDB JSON document:

1) {}
2) { "user_id"=1,
   "user_name"="Joe Sanders",
   "occupation"=["engineer","writer"]
  }
3) { "user_id":1;
   "user_name":"Joe Sanders";
   "occupation":["engineer","writer"]
  }
4) { "user_id":1,
   "user_name":"Joe Sanders",
   "occupation":["occupation1":"engineer", "occupation2":"writer"]
  }

Q10) Which of the following commands will return all the posts with number of likes greater than 100 and less than 200, both inclusive?

1) db.posts.find({ likes : { $gt : 100, $lt : 200 } });
2) db.posts.find({ likes : { $gte : 100, $lt : 200 } });
3) db.posts.find({ likes : { $gt : 100 , $lte : 200 } });
4) db.posts.find({ likes : { $gte : 100 , $lte : 200 } });

Student Work Area

Algorithm/Flowchart/Code/Sample Outputs

EXPERIMENT NO. 10

Student Name and Roll Number:
Semester /Section:
Link to Code:
Date:
Faculty Signature:
Marks:

Objective(s): Retrieve data from NoSQL database - MongoDB.
Outcome: The students will be able to perform advanced CRUD operations in MongoDB.
Problem Statement: Create a database with the following collections: **Project** Possible Fields: • Project Name (Pname) • Project Number (Pnumber) • Department Number in which project is being run (Dnum) • Project Locations (Plocation) ➢ Plocation can be an array of multiple locations (or) ➢ Plocation can be a composite field with sub-fields – city, state, country

Department

Possible Fields:

- Department Number (Dno)
- Department Name (Dname)
- Department Manager (Dmanager)

Execute the following queries:

1) Insert multiple documents (2-3) together in the Project collection with atleast 1 project location as a composite attribute

2) Write the “Select * from Project” equivalent query in MongoDB.

3) Write a MongoDB query to update the project location of all projects to “Pune” and department number to “20” with project name “ProjectX”.

4) Write a MongoDB query to delete all the documents of collection “Project” with department number “4”.

5) Write a MongoDB query to retrieve the project details of all projects with project location as “Delhi” and “Mumbai” both assuming the following documents exist.

   { "_id" : ObjectId("5cc7fe36e37496f1a5e7e268"),"Pname" : "ProductK", "Pnumber" : 45,"Plocation" : ["Delhi","Mumbai","Pune"], "Dnum" : 3}
   { "_id" : ObjectId("5cc7fe36e37496f1a5e7e268"),"Pname" : "ProductT", "Pnumber" : 56,"Plocation" : ["Delhi","Mumbai"], "Dnum" : 15}
   { "_id" : ObjectId("5cc7fe36e37496f1a5e7e268"),"Pname" : "ProductU", "Pnumber" : 5,"Plocation" : "Delhi", "Dnum" : 6}
   { "_id" : ObjectId("5cc7fe36e37496f1a5e7e268"),"Pname" : "ProductL", "Pnumber" : 4,"Plocation" : "Mumbai", "Dnum" : 7}

6) Write a MongoDB query creating a One to Many Relation between a document of “Project” Collection and “Department” Collection.

7) Write a MongoDB query to retrieve the project details → project name, department number, department name and department manager of projects with department manager (for department in which the project is being run) as “ABC”.

Background Study:

1) pretty() is used to display the query obtained in a neat manner with all the fields clearly depicted one below the other.

2) Data for a particular field can also be stored as an array with multiple values for the field.

3) $all is used to match the array items for the particular field.
   - *db.project.find({Plocation:{$all:["Delhi","Mumbai"]}}).pretty()*

4) $elemMatch: This operator matches documents that contain an array field with at least one element that matches all the specified query criteria.
   - Syntax: { <field>: { $elemMatch: { <query1>, <query2>, ... } } }

5) Fields can be embedded in the field too and while using find in such a case, always refer the field embedded in another field by using a dot notation and in quotes.

6) AND/OR Operations

Operation	Syntax	Example
AND	>db.collection_name.find({$and: [{key1: value1}, {key2:value2}]})	>db.project.find ({$and:[{Dnum:5}, {Pname:"ProductZ"}]}).pretty();
OR	>db.collection_name.find({$or: [{key1: value1}, {key2:value2}]})	>db.project.find ({$or:[{Dnum:5}, {Pname:"ProductZ"}]}).pretty();

7) Documents can either be embedded in one to one relation or one to many relation
   - In one to one, the new field contains only a single embedded document.
   - In one to many, the new field contains an array of multiple embedded documents.

Question Bank:

Q1) The order of documents returned by a query is not defined unless you specify a _____

1) sortfind()
2) sortelse()
3) sort()
4) none of the mentioned

Q2) In aggregation pipeline, the _____ pipeline stage provides access to MongoDB queries.

1) $catch
2) $match
3) $batch
4) All of the mentioned

Q3) Point out the wrong statement.

1) sort() modifier sorts the results by age in ascending order
2) Queries in MongoDB return all fields in all matching documents by default
3) To scale the amount of data that MongoDB sends to applications, include a projection in the queries.
4) None of the mentioned

Q4) Which of the following query selects documents in the records collection that match the condition { “user_id”: { $lt: 42 } }?

1) db.records.findOne({ “user_id”: { $lt: 42 } }, { “history”: 0 })
2) db.records.find({ “user_id”: { $lt: 42 } }, { “history”: 0 })
3) db.records.findOne({ “user_id”: { $lt: 42 } }, { “history”: 1 })
4) db.records.select({ “user_id”: { $lt: 42 } }, { “history”: 0 })

Q5) Which of the following is not a projection operator?

1) $slice
2) $elemMatch
3) $
4) None of the mentioned

Q6) _____ is a client or database-generated identifier that uniquely identifies this message.

1) messageLength
2) responseTo
3) requestID
4) all of the mentioned

Q7) When inserts, updates and deletes have a _____ write concern, write operations return quickly.

1) strong

2) weak
3) average
4) very strong

Q8) Which of the following MongoDB query is equivalent to the following SQL query:

UPDATE users SET status = "C" WHERE age > 25

1)

```
db.users.update(
  { age: { $gt: 25 } },  { status: "C" })
```

2)

```
db.users.update(
 { age: { $gt: 25 } },
 { $set: { status: "C" } })
```

3)

```
db.users.update(
 { age: { $gt: 25 } },
 { $set: { status: "C" } },
 { multi: true })
```

4)

```
db.users.update(
 { age: { $gt: 25 } },
 { status: "C" },
 { multi: true })
```

Q9) Consider that you have a collection called population which has fields state and city. Which of the following query will calculate the population grouped by state and city?

1) db.population.aggregate([{ $group: { _id: { state: "$state", city: "$city" },pop: { $sum: "$pop" } } }])

2) db.population.aggregate([{ $group: { _id: { state: "$state", city: "$city" },pop: { $sum: 1 } } }])

3) db.population.aggregate([{ $group: { _id: { state: "$state", city: "$city" },pop: { “$pop”: 1 } } }])

4) db.population.aggregate([{ $group: { _id: { city: "$city" },pop: { $sum: "$pop" } } }])

Q10) Which of the following command is used to get all the indexes on a collection?

1) db.collection.getIndexes()
2) db.collection.showIndexes()
3) db.collection.findIndexes()
4) db.showIndexes()

Student Work Area

Algorithm/Flowchart/Code/Sample Outputs