Format: **45 Multiple Choice**

5 solutions: 2 normalization skills
1 ERD
2 SQL examples: create table
Select query

ERD

⇒ Conceptual view (abstract)
⇒ Top-down approach – macro view
⇒ Be able to identify & construct key components
  - Entity
  - Relationship
  - Connectivity
  - Cardinality
  - Bridge entity
  - Weak entity

⇒ Approaches
  - Chen, Crow's foot, etc…

⇒ Works in complimentary fashion with normalization
⇒ Focus is entity definition and the proof that they belong in database
  - Proof is the relationship

⇒ Rules based on model
  - M:N OK at first but not allowed in Relational model

⇒ Types of relations
  - Unary
  - Binary
  - Ternary

Normalization

⇒ Goal is to control redundancy
  - Existence of relationships makes it impossible to eliminate

⇒ Focus is table definition and construction
  - Entities breed tables
  - Tables subsequently create additional relationships

⇒ Micro view of database
⇒ Bottom-up approach (detailed to abstract)
⇒ Guidelines
  - 1NF
    - First normal form identifies the key(s)
      - Prime attributes = keys
      - Non-prime attributes = non key values

        - Creates dependency diagram to show the dependencies
            - Transitive & partial
        - In keys: eliminate nulls & repeating groups
    - 2NF
        - Only applies when there is a composite key
        - Goal here is to eliminate partial dependencies
            - Typically results in creating new tables
    - 3NF
        - Realistic end point in traditional normalization
        - Elimination of transitive dependencies
            - Dependencies of non-prime to non-prime
        - Look to resolve issues with atomicity & derived attributes
        - Traditional end point of de-normalization
    - BCNF – 4NF – 5NF
        - Largely theoretical
        - Database purist would go here but due to issues with performance and practical use
            - De-normalize back to 3NF

⇒ Finalize table structure
- Define attributes
    - Naming conventions
    - Atomicity
    - Derived?
    - Data types
        - CHAR
        - VARCHAR
        - INTEGER
        - NUMERIC
        - DATE
    - Define keys

SQL

⇒ Database language used to physically create & maintain the database
⇒ 4th generation language
- Non-procedural

⇒ Categories of SQL
- DDL
    - Data definition used to create tables & indexes
        - Index used to facilitate searching and selection
        - Table of values and pointers
            - Potential overhead
    - Primary Key create
    - Foreign key creation is in table

- Provides the basis for relationship
- CASCADE constraint

- DML
  - Basis for data manipulation
  - Rules of precedence in arithmetic calculations
  - Common commands
    - Insert
    - Join
    - Update
    - Select
      - Where – Order By
      - LIKE (partial Where – Wildcard -- % or *)
- DCL
  - Data control, used for security and administration

⇒ Dialects
- SQL has an ANSI standard but there are differences among the various DBMS's

⇒ Syntax
- Begins with SQL command/keyword and ends with semi-colon ;

⇒ Examples

- Create table

```
CREATE TABLE ccri_std (
c_ID char(4) not null,
c_lname varchar(20) not null,
c_fname varchar(15) not null,
c_city varchar(20),
c_state char(2),
c_billamt currency,
c_billDT date,
Primary key (c_ID));
```

- Select

```
SELECT * FROM employee
ORDER BY salary DESC;

SELECT name, salary from employee
WHERE emp_type = "G";
```

- Join

```
SELECT Client.ClientName, Booking.TripDate,
Booking.People
FROM Client INNER JOIN Booking ON
Client.[Client#] = Booking.[Client#];
```