Database Cheat Sheet

By Ammar Jaber

Table of Content

Introduction

DataBase: Collection of related data.

Database Managment System(DBMS): a collection of programs that enable users to create and maintain a database.

Schema: description of the database.

Row in a table = tuple = record = instance = occurrence.

DBMS Architectures:

1. Centralized Architecture:
    - Computer = DBMS + DB + APPLICATION.
2. 2-tier Architecture:
    - Server: DBMS + DB
    - Computer: Application.
3. 3-tier Architecture:
    - Database Server: DBMS + DB.
    - Web server: Application.
    - Client: Web Browser.

Entity Relation Diagram (ERD)

1. **Entity**: it will be mapped to table later.
2. **Attribute**: it will be mapped to table column later.

Simple: has one value	**Composite**: has sub attributes
Single: one value	**Multi**: more than one value
Stored: we get this value from user	**Derived**: we derived it from another attr

3. **Relationship:**
    a. Binary.
    b. Ternary.
    c. N-ary.
    d. Recursive.

Constraints on Relationship:

1. **Cardinality Ratio**
    a. 1:1
    b. 1:N
    c. N:M
2. **Participation**
    a. Total
    b. Partial
3. **Min,Max notation**

- **Note**: Weak entity key = partial key + strong entity key
    - **Partial key**: some attributes of weak entity.
    - **Strong entity key**: key of strong entity that defines weak entity.

Enhanced ER Diagram (EERD)

Enhanced ER = ER + Enhancement

Enhancement:

- Subclasses, superclasses, and inheritance.
- Specialization and generalization.
- Multiple Inheritance (shared subclasses).

Superclass: contain all the common attributes for the subclasses.

Subclasses: contain special attributes for each subclass

Specialization: defining subclasses for an entity.

Generalization: reverse process of specialization.

Constraints no specialization and generalization:

1. **Based on definition**
    a. **Predicate defined subclasses**: depend on superclass attribute we know in which subclass should go.
    b. **User defined subclasses:**
2. **Based on membership**
    a. Disjoint: new instance should go to only one subclass
    b. Overlapping: new instance could go to more than on subclass
3. **Based on participation:**
    a. Total:
    b. partial:

Note: if the specialization predicate defined that does not mean that the membership should be disjoint.

Mapping

Mapping: ER and **EER** diagrams to **tables(relations).**

Mapping ER Diragram:

1. **Regular entity (strong)** : if we have composite attribute we take the parts and leave the parent.
2. **Weak entity:** foreign key + partial key = strong key
3. **Binary (1 : 1):** we take the foreign key from the first to second and vice versa.
4. **Binary (1 : N):** take the primary key from **1** and put it as foreign key in **N.**
5. **Binary (N : M):** we create a new table and we take each foreign key from **N** and **M** and both of them are the primary key.
6. **Binary N-ary:** same as many to many relationship
7. **Multi-value attributes:** new table with foreign key from the prev table

Note: the relationship attribute follows the foreign key where ever he goes.

Note: derived attributes are not anclouded in mapping.

Note: int one to one relationship we put the foreign key in the total side not partial because partial reduce null values.

Mapping EER Diragram:

1. **Option 1**: Multiple relations, superclass and subclasses
    a. Works for **any type** of specialization
    b. البرايمري كي من السوبر لكل جدول في ال سب كلاسيس
2. **Option 2:** Multiple relations, subclasses only.
    a. Because all the attributes lands from the superclass to the subclasses
    b. Works only for **total** specialization
3. **Option 3**: single relation
    a. We create type attribute because all data in one table
    b. Works only for **disjoint** specialization
    c. Generates lots of **null values**
4. **Option 4:** single relation
    a. Each subclass have type attribute
    b. Works for overlapping and disjoint

Note: mapping shared subclasses choose one of the 4 options.

Relational Algebra

ER Diagram + EER Diagram → Relations (**tables**).

Relational Algebra → **Queries.**

Unary relational operators:

- **Project(Π)**: vertical selection.
- **Select(σ)**: Horizontal selection.
- **Rename(ρ):**
    - Rename **relation + attributes**.
    - Rename only **relations**.
    - Rename only **attributes**.

Binary Relational Operators:

- **Union (U):** we merge the two Relations (tables) and **remove the duplicates.**
- **Intersection (∩):** we take from the two Relations (tables) the **similars.**
- **Set Difference:** R1 - R2 == exist in R1 but not ∩ with R2 == R1 - R1 ∩ R2
- **Cartesian Product:** we merge each row from R1 with each row from R2

  number of rows = R1 tuples * R2 tuples

  Number of columns = R1 tuples + R2 tuples
- **Join:** Binary(cartesian X) then Unary (select σ)
- **Division:** R1 row connected with all R2 rows
- **Aggregate:** Operations like sum

Question: on unary and binary relational operators

employee

FirstName	LastName	DepartmentNumber	Salary
Ammar	Jaber	1	1,200
Ahmad	Shraim	2	900
Baraa	Mohammed	3	700

Example(1): give me **first name** and **last name columns** from **employee** table

Answer:

R = Π (FirstName,LastName) employee

FirstName	LastName
Ammar	Jaber
Ahmad	Shraim
Baraa	Mohammed

Example(2): display employee who work in dep 1 with first name, last name and salary

Answer:

R1 = σ (DepartmentNumber = 1) employee

R2 = Π (FirstName,LastName,Salary) R1

FirstName	LastName	Salary
Ammar	Jaber	1,200

Example(3): show first name of employee with salary less than 1,000 and grater than 500

Answer:

R1 = σ (salary < 1,000 **and** salary > 500) employee

R2 = Π (FirstName) R1

FirstName
Ahmad

Example(4): rename table name from **employee** to emp

Answer:

R1 = ρ **employee**

emp

Example(4): rename table name from **emp** to employee and FirstName to FN

Answer:

R1 = ρ **emp**

employee(FN**)**

Normalization

Functional Dependencies and Normalization for Relational Databases

Perfect Relations should be devoid of:

1. **Redundantly**
    a. Wastes storage
2. **Update anomalies**
    a. Insertion
    b. Deletion
    c. Modification

- ال Relation المثالية هي التي تخلو من عيب التكرار Redundancy والعيوب الناتجة عنه في عملية تحديث البيانات سواء في (insert - update - delete)

Normalization: the process of decomposing unsatiisfaory “bad” relations by breaking up their attributes into smaller relations.

Normal Form: Condition using keys and FDs of relation to certify whether a relatin schema is

X -> Y : x and y are attributes and the arrow is a FD(functional dependency)

- Y functionally dependent on X
- X determinants Y

Example:

SSN -> Name

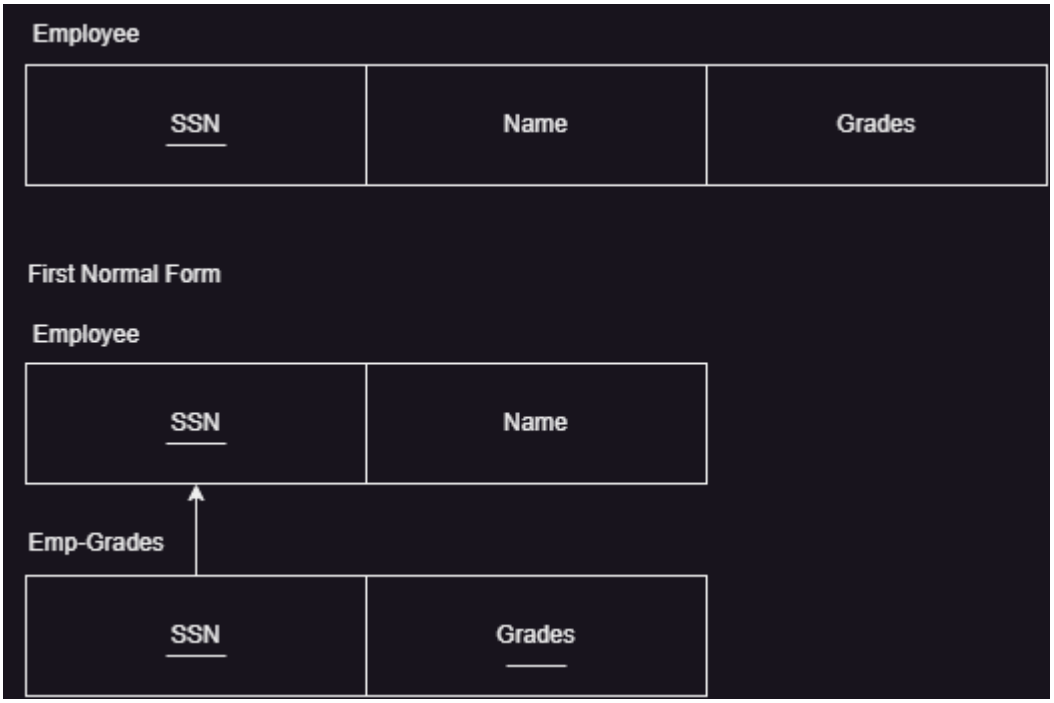
- Ssn **detement** name
- Name **dependent** ssn

SSN, PNumber -> Hours

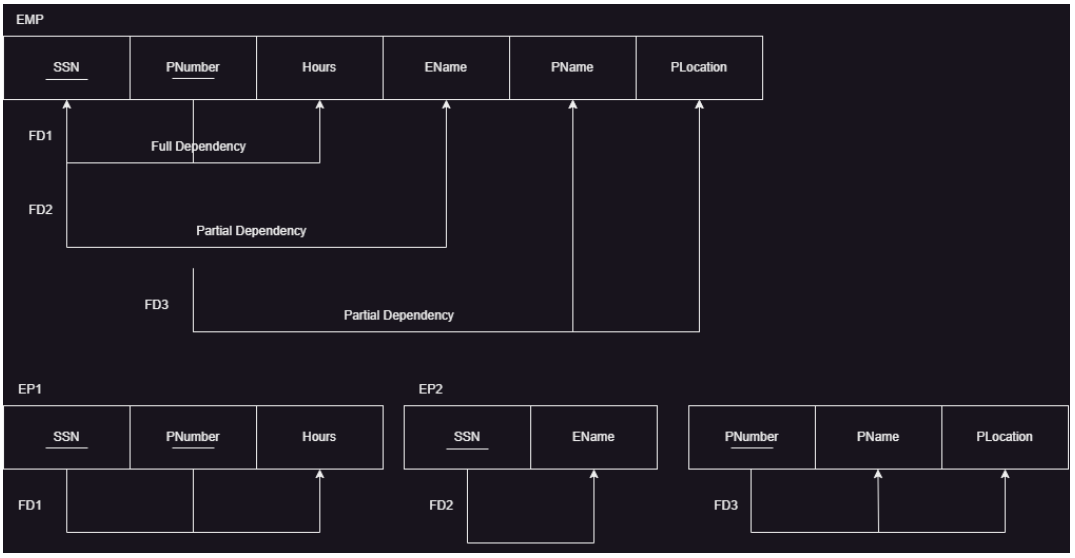
- ال **Functional Dependencies** هي علاقة او ارتباط بين attribute و attribute اخر في نفس الجدول.

Normal Forms Based on Primary Keys

1. First Normal Form
    - Disallows Composite attributes
    - Disallows Multivalued attributes
    - Disallows Nested relations



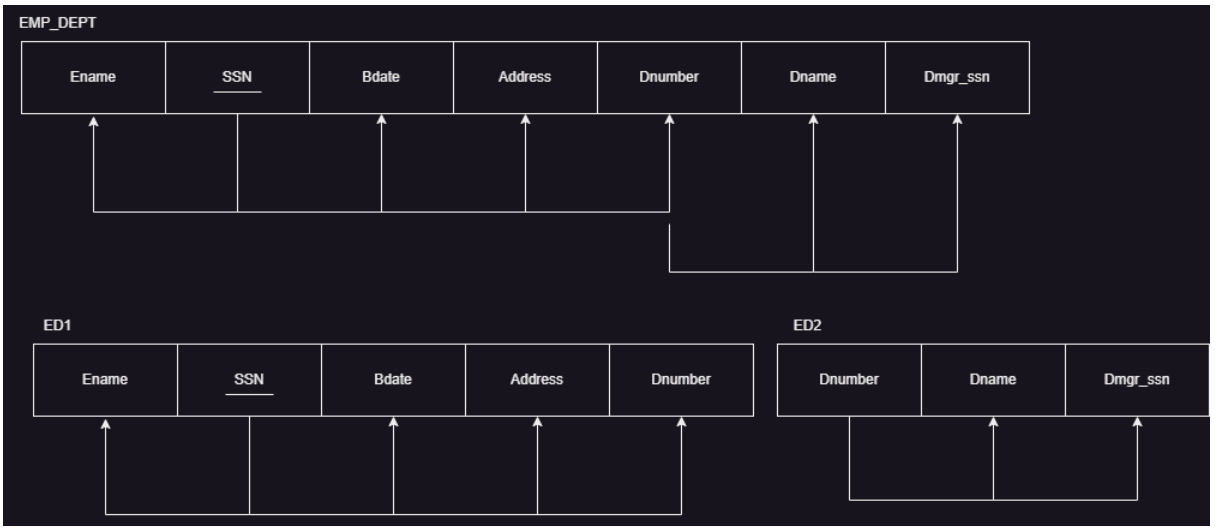
2. Second Normal Form:
    - Disallows Partial Dependency: normal attribute is dependent on part of the pk.
    - Be in 1NF



3. Third Normal Form.

حتى يصبح الجدول في ال 3rd NF يجب ان يكون بالفعل في ال 2ND NF وان يخلو من مشكلة ال Transitive Dependency

- Be in 2ND NF
- Disallows Transitive Dependency: **which means that a non key attribute is dependent on another non key attribute.**



4. Boyce Codd Normal Form.
    - To table be BCNF every attribute should depend on super key even if it was the prime key
5. Fourth Normal Form.
6. Fifth Normal Form.

Closuer

Example(1):

$A \rightarrow B$

$B \rightarrow D$

$C \rightarrow DE$

$CD \rightarrow AB$

Answer:

A+ = {A,B,D}

B+ - {B,D}

C+ = {C,D,E,A,B}

D+ = {D}

E+ = {E}

CD+ = {C,D,E,A,B}

AD+ = {A,D,B}

ADB+ = {A,D,B}