Week#3

Basic Query Formulation

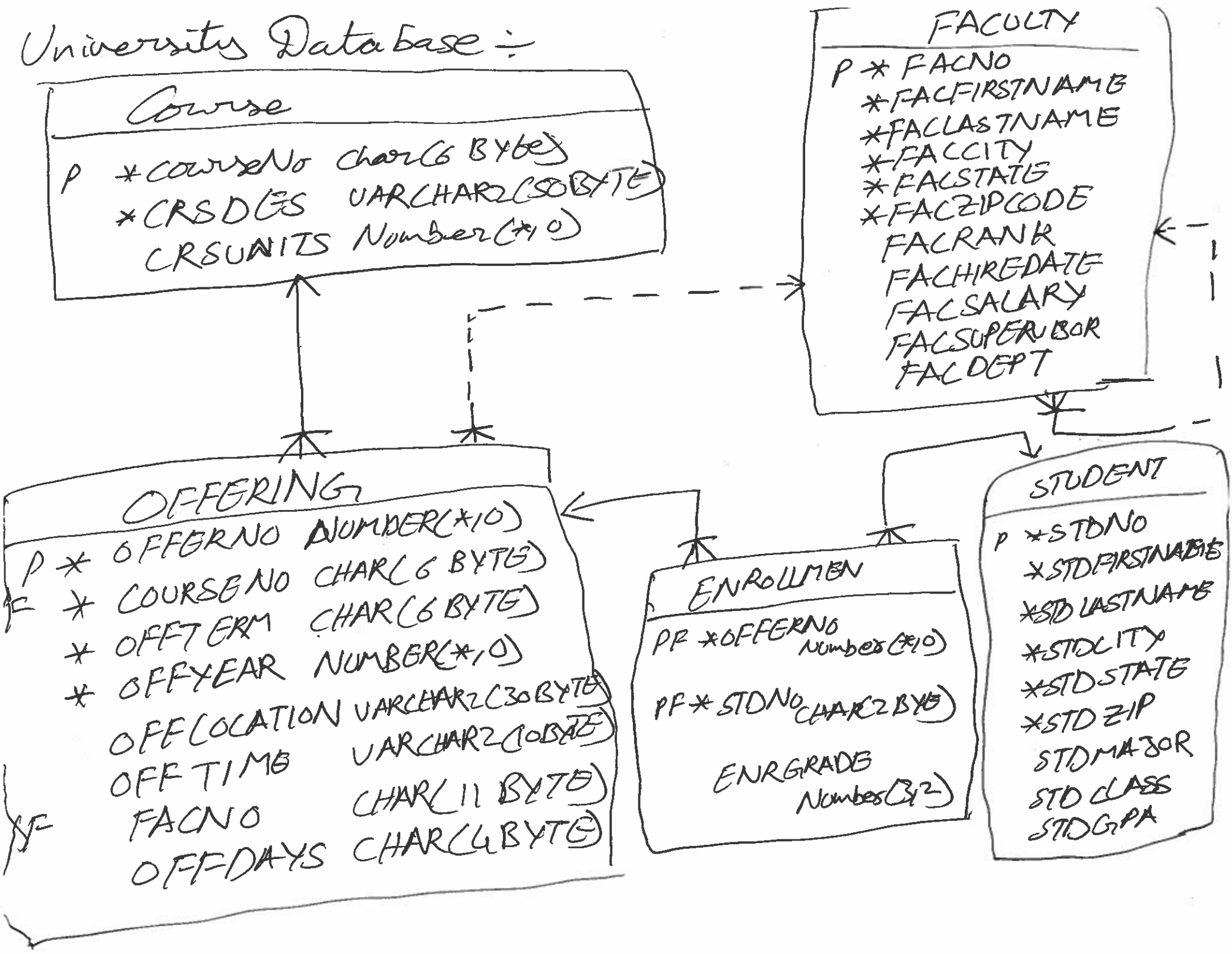
SQL :-
Structured query language.

SQL has 3 types of statements :-
Definition
Control
Manipulation

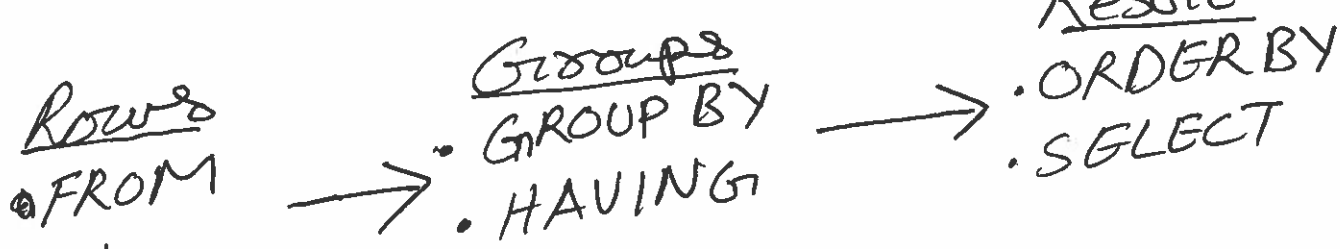
SQL
Conformance issues reduced portability across DBMS's & higher switching costs.
→ SQL is a broad language with weak conformance.
→ for single table problems, you must know the columns of a table & data types.

Join Operators :-
Used to combine tables in Databases.
→ Equi-Join (when join operator involves "=")

University Database :-

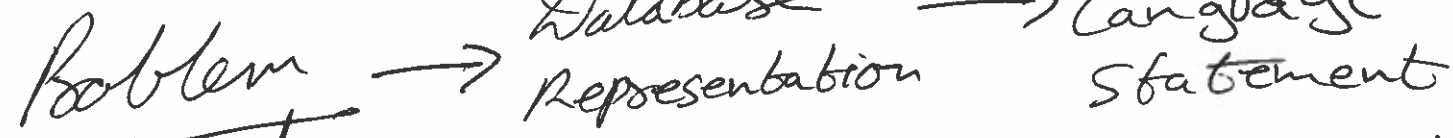


Query Clause EVALUATION ORDER:-



Row operations before grouping operations
Grouping only occurs one time

Query Formulation Process:-



Aspects to consider before formulating queries:-

→ Databases & tables neccessary
→ combination of ~~tables~~ columns to link b/w the tables.
→ aggregation neccessary group of rows (or) individual rows.

Efficiency Considerations:-

- No extra tables
- No unneccessary grouping
- No missing join conditions.

Set operations :-

Union :- $A \cup B = A + B - (A \cap B)$ ⇒ Union

Intersection :- $A \cap B$ ⇒ $e \in A \& e \in B$ ⇒ INTERSECT

Subtraction :- $A - B = e \in A \& e \notin B$ ⇒ MINUS

MODIFICATION STATEMENTS in SQL :-

Typical Modification statements are :-

- ⇒ Insert
- ⇒ Update
- ⇒ Delete

No typically used has many database use custom data entry forms & Backend modification are typically code base inserts.

Syntax :

⇒ INSERT INTO TABLE table_name VALUES ('s601', Num, Date, 'S602'), (---);

→ UPDATE table_name SET
col_name_1 = (x y),
col_name_2 = (z z)
where col_name_3 = (y y)
AND col_name_1 = (x x');

Update is not possible on Reference columns like Primary Key

→ DELETE FROM table_name
where col1_name = (x x)
AND col2_name = 'x y';

Typicall rows in a Child Table should be deleted before associated rows in parent tables.

Query Formulation Errors :-

Types of Errors:

- Syntax
  - occurs in code
  - Detection before execution
- Redundancy
  - Excessive resource usag
  - to many unnecessary tables & Columns involved.
- Semantic
  - ~~Runtime error~~ Incorrect result due to failed logic
  - Missing row condition, col-name ex) parentheses.
- Runtime errors
  - platform based errors.

Week #4

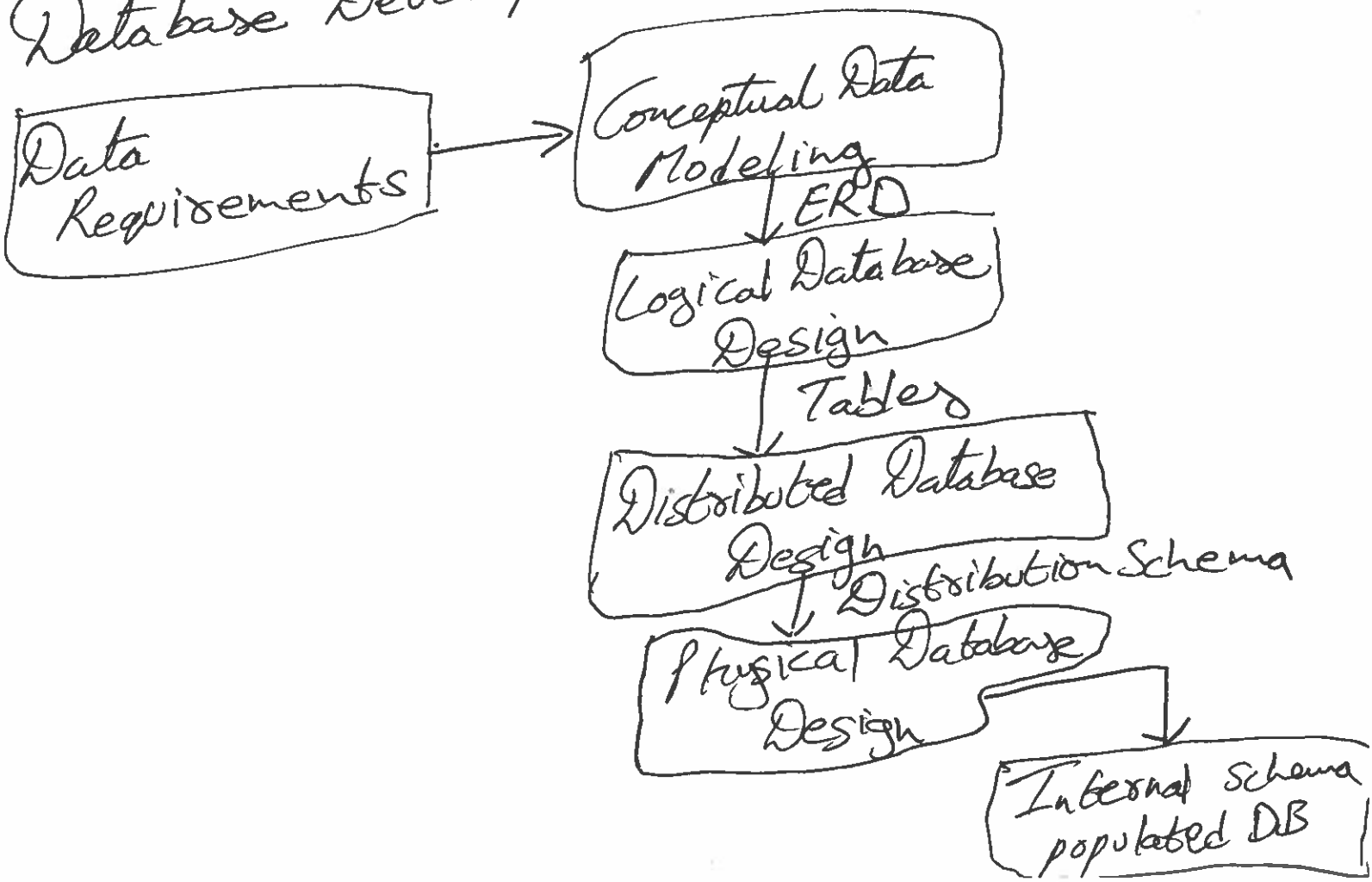
Notation for Entity Relationship Diagrams

Database Development goals :-

Board goals of Database Development:

- A common vocabulary for Data Dictionary
- Define business rules
- Ensure data quality
- Provide efficient implementation.

Database Development phases :-



ERD Notation:-

ERD consists of 3 components:-

- Entity types
- Relationships &
- Attributes

ERD Differes from Relational Database Diagra in the aspects of:

Relational Database Diagram Donot use name for relationships instead mark the attributes has Primary key & Foriegn k

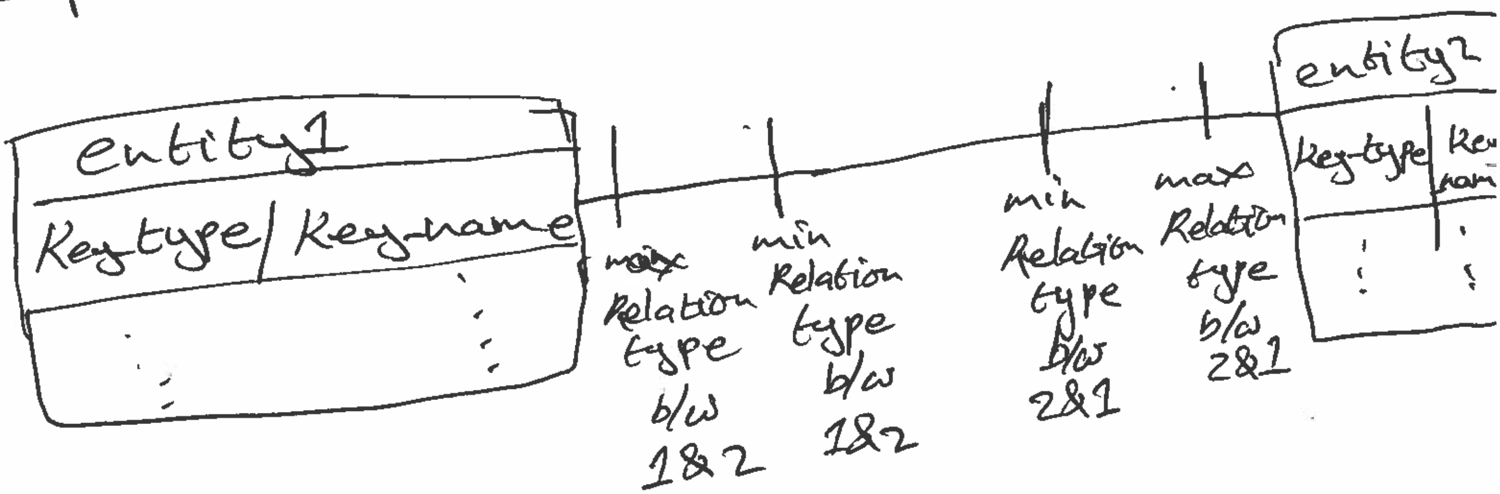
Relationship variations:-

Weak entity type:
borrows parts or entire Primary key

Identifying relationship:
Provides a component of a Primary key for a weak entity type.

Identification dependency:
A weak entity type & one or more identifyin relationships

Sample ERD :-



- Relation type can be | (for one), 0 (for zero) & < (for many).
- Key-type can be PK (primary key) & FK (Foreign key). & ~~UK (Unique Key)~~
- Optionally can also include Datatype & Constriant type(& int, varchar, string etc...) & Unique, not null, checksum {>50, etc...}).

Week #5

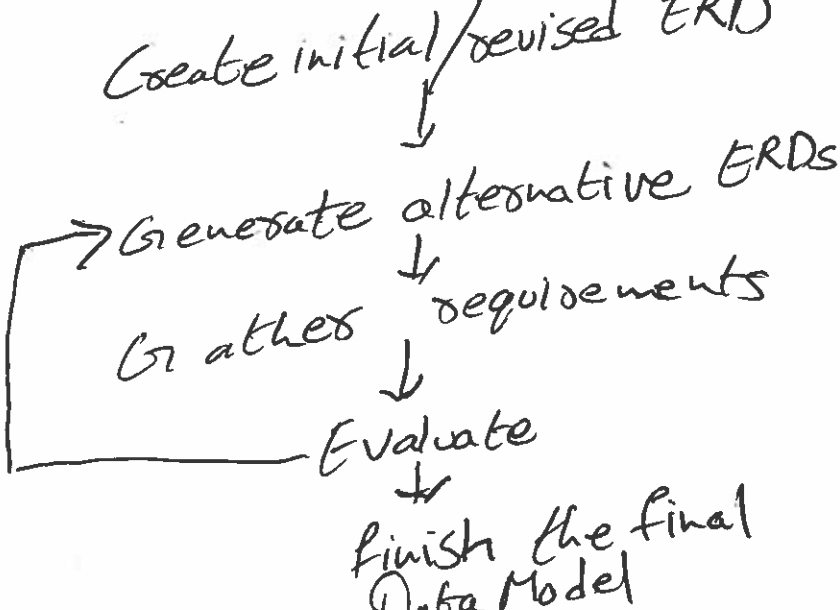
Developing Business Data Models

Conceptual data Modeling goals & challanges:-

- Analyzing narrative problems
  - Process:-
    - Consistency with narrative
    - Identify deficiencies
    - Prefer simpler designs
    - Identify potential refinements
  - solution:-
    - Identify entity types & attributes
    - Determine primary keys based on stability & purpose.
    - Identify entity & attribute relationships

Data Modelling Problems & Completion of an ERD

ERD Refinement Cycle:-



Week#6

Schema Conversion

Goals of logical Database Design:-

Managing Redundancy

logical Database Design:-

Convert + Specify → Normalize → Refine

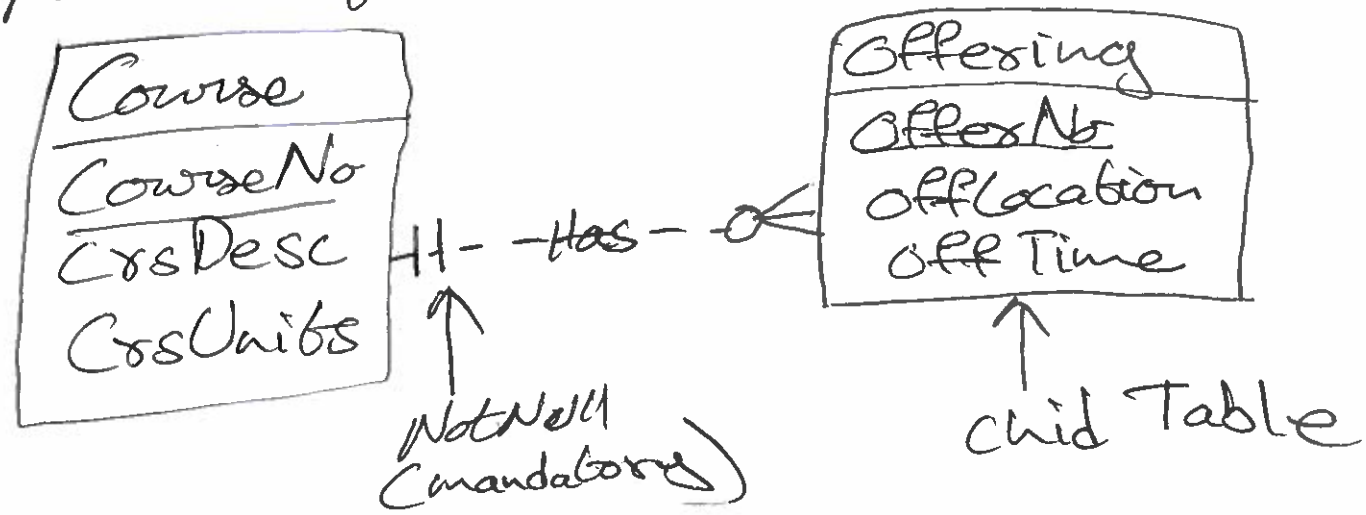
- ERD to intial table design
- Redundancy Constraints
- Eliminate unwanted redundacies
- Uniqueness Constraints
- other Constraints

Conversion Rule:-

Entity Type Rule → 1-M relationship rule → M-N relationship rule → Identifying relationship rule

- * Identify Table
- * FK's in the child table
- * Associative table plus FKs
- * Combined PK
- * Components of PK

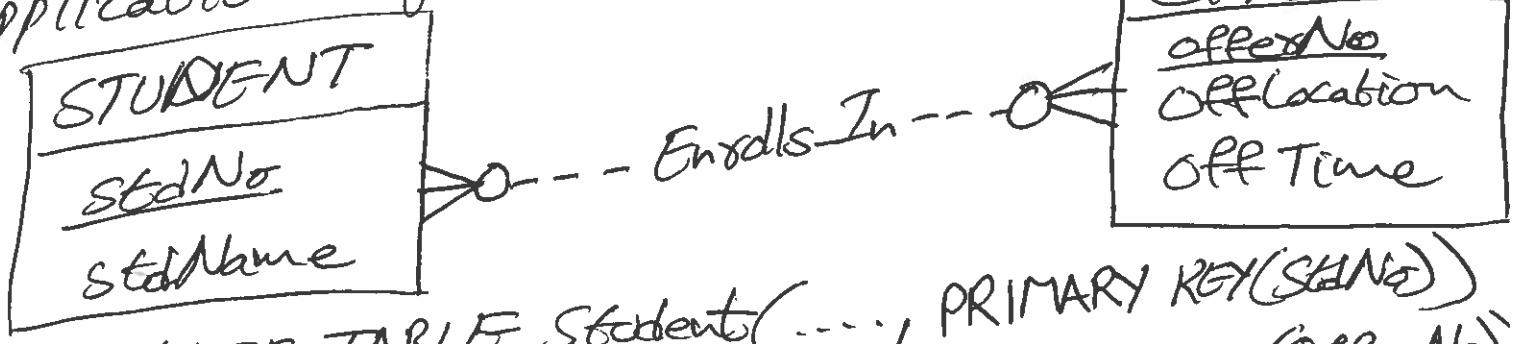
Application of Basic Rules I :- (1 to Many)



CREATE TABLE Course(..., PRIMARY KEY (CourseNo))

⇒ CREATE TABLE Offering(..., PRIMARY KEY (OffrNo), FOREIGN KEY (CourseNo) REFERENCES Course, CONSTRAINT CourseNo NOT NULL)

Application of Basic Rules II :- (Many to Many)



CREATE TABLE Student(..., PRIMARY KEY(StdNo))
CREATE TABLE Offering(..., PRIMARY KEY(OffrNo))
CREATE TABLE Enrollment(..., PRIMARY KEY(StdNo, OffrNo), FOREIGN KEY(StdNo) REFERENCES Student, FOREIGN KEY (OffrNo) REFERENCES Offering)

Strong vs Weak relationship in ERD :-

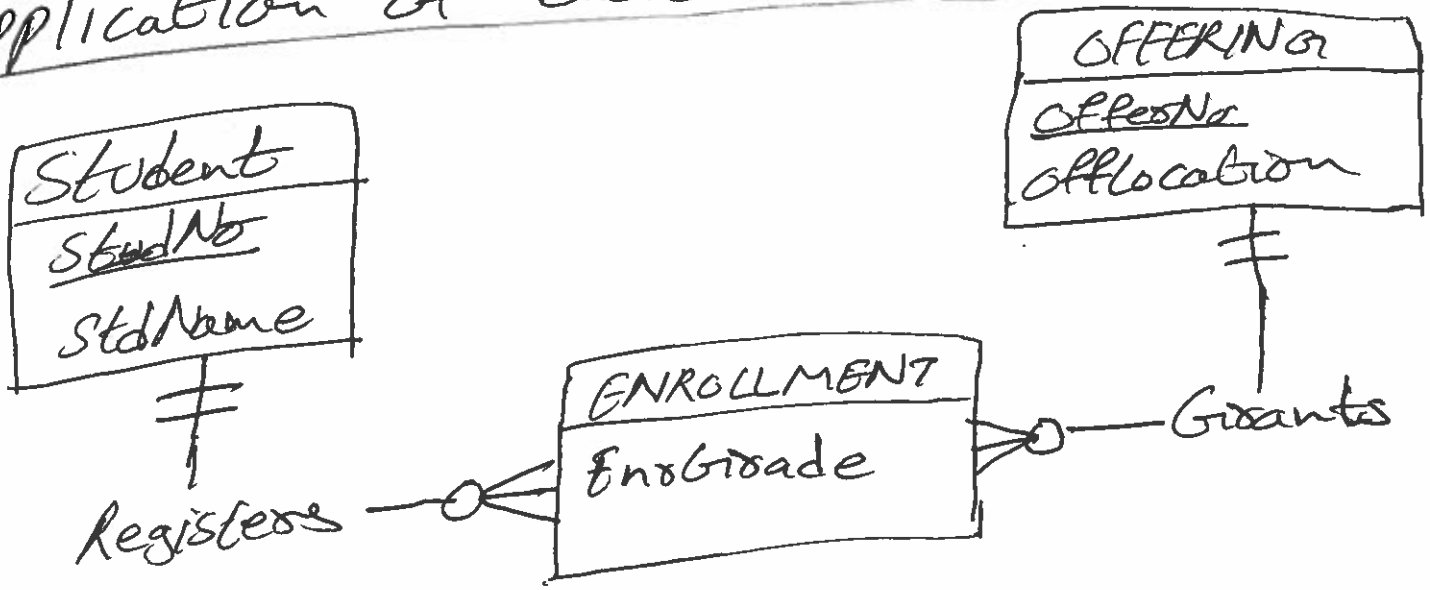
Weak (Non-Identifying) Relationships

- Entity is existence-independent of other entities.
- PK of child doesn't contain PK component of parent Entity.

Strong (Identifying) Relationships

- Child entity is existence-dependent on parent.
- PK of child Entity contains PK component of parent Entity.
- Usually occurs utilizing a composite key for primary-key which means one of this composite key components must be primary key of the parent entity.

Application of Basic Rules III



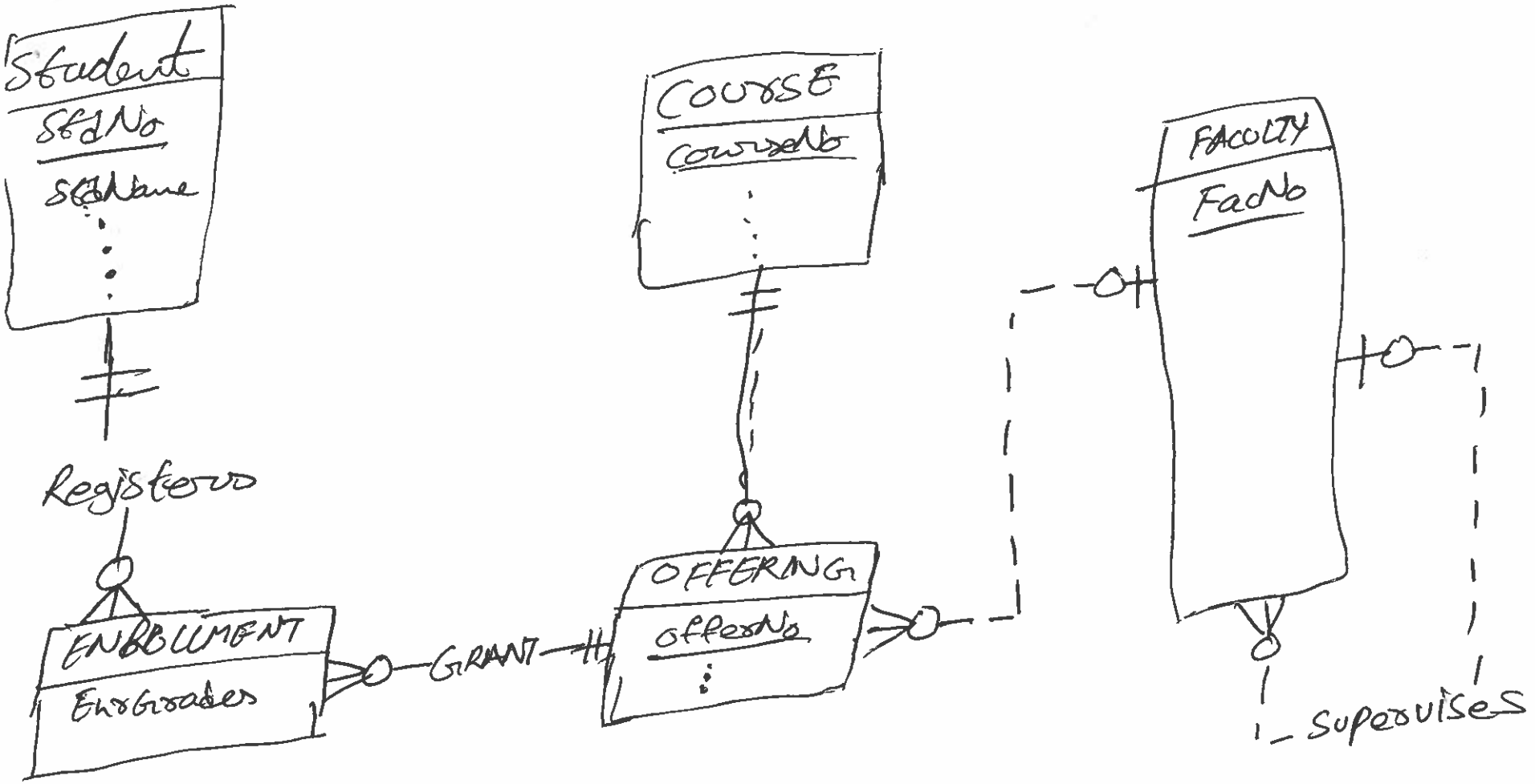
Conversion :-

→ Entity type rule is applied 3 times for student, Enrollment, offering.

→ 1-M relationship rule is applied twice to create FK's student number & offersNo.

→ Because of Weak relationship due to Identification dependency rule both "OfferNo" & "StdNo" are added to the "Enrollment" table.

University Database ERD:



University Database Conversion:-

Entity type rule ⇒ • course • Faculty • Student • Offering • Enrollment

table

1-M relationship rule
(FK's for child tables from Parent tables)

⇒ Enrollment.StdNo
Enrollment.OffesNo
Offering.CourseNo
Offering.FacNo
Faculty.FacSupNo
↓
(added due to self referencing entity in Faculty)

M-N relationship rule
(Create new table for the relationship b/w entities & FK's & PK's from both entities into the relationship table.)

⇒ No application

Identifying relationship rule
(borrows PK's from dependent entities as PK's)

Week #7

Normalization Concepts & Practice

Modification anomaly is an unexpected side effects from a row operation. Typically Modification anomalies typically occur due to poor Table Desig, where an operation for a set of related Tuples can also effect the ~~ope~~ unrelated set of tuples (example might be a single table Design instead for 5 far university databases.) & Bai Transaction processi is more sensitive to modification anamolies than B.I processing

Pros vs Cons :-

Single Tabular Data	multiple Tabeled Data
→ easy to query. due to No join operations & reduced computational power.	→ join operation required hence more problem with querying & computing.
→ less security & Data validity in case of Modification anamolies.	→ More security & Data validity in case of Modification anamolies.

Functional dependencies :-

- Functional dependency is a value neutral constraint similar to primary key & FK's domains. It involves a comparision of columns.
- Specification of functional dependies drives the normalization process.
- FD is a constraint about two or more columns of a table.

⇒ X determines Y ⇒ X → Y ⇒ for each X value, there is atmost one Y value.

Ex :- stdNo → stdCity if each stdNo value has atmost one stdCity value.

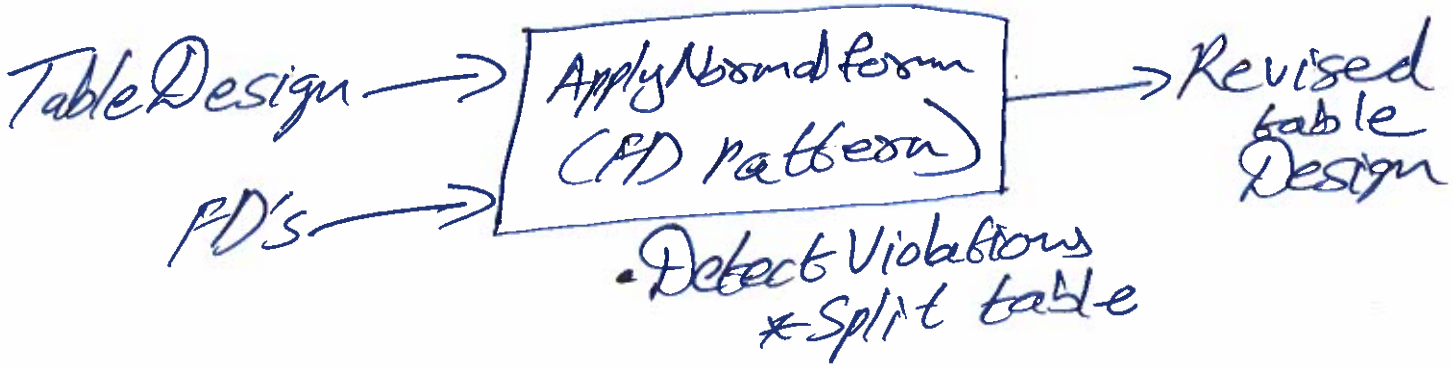
similar to uniqueness constraint.

* Columns with FD should be placed in the same table.

Normalization :-

Normalization is the process of removing unwanted redundancy in a table design using Functional dependencies (FD's).

Normalization process :-



BOYCE-CODD Normal Form (BCNF) :-

Simple Definition of FD's

↓

check if Determinant is unique or not

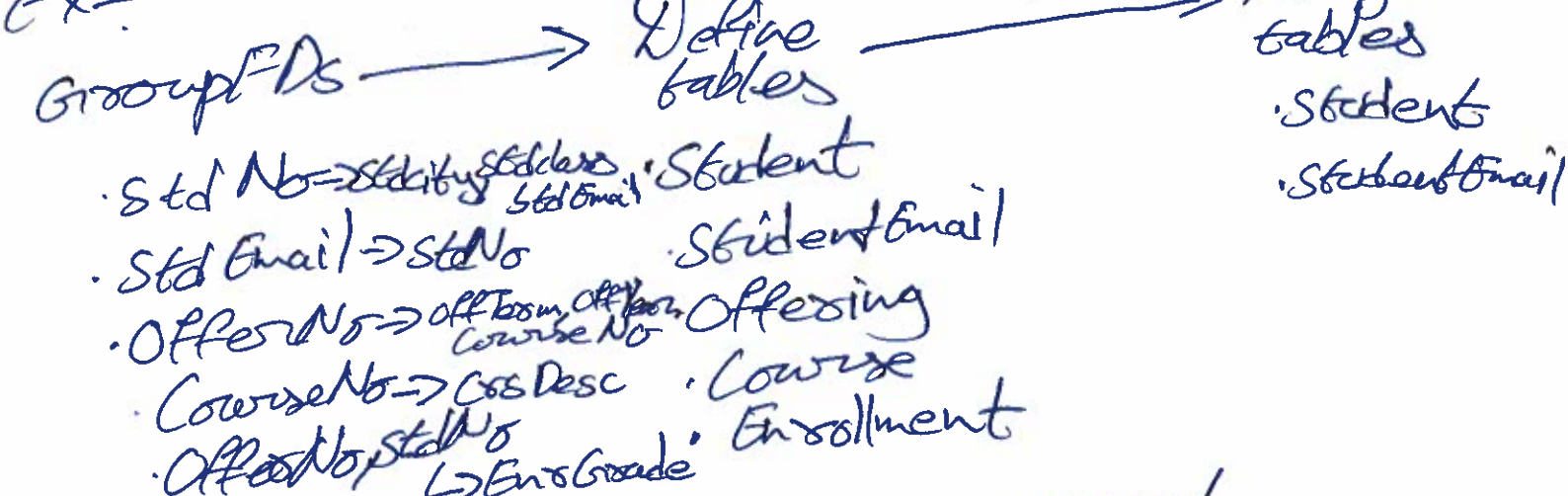
↓

Apply BCNF

Ex :-

Or

Ex :-



Student & StudentEmail were merged.

offering & course were merged.