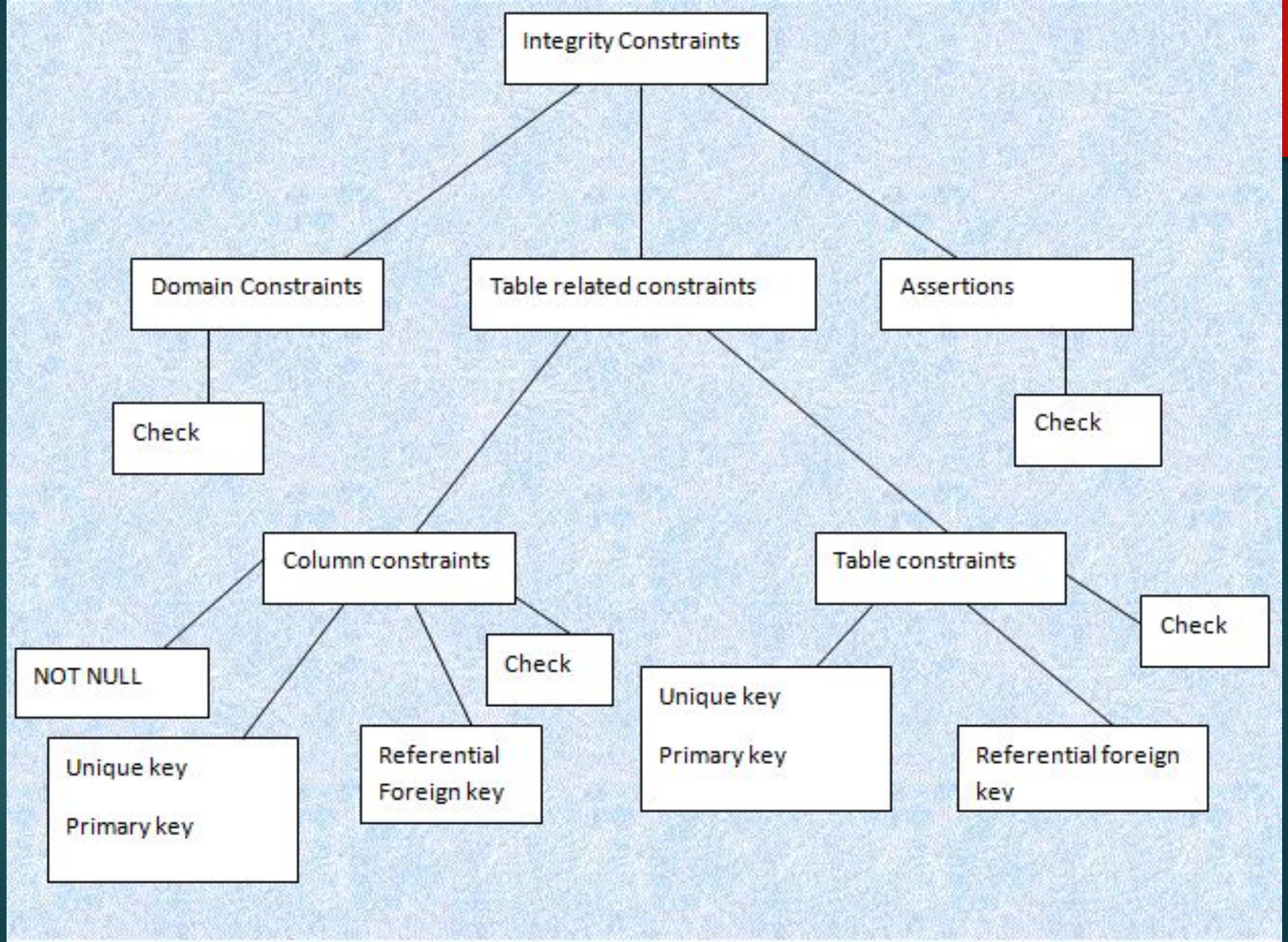
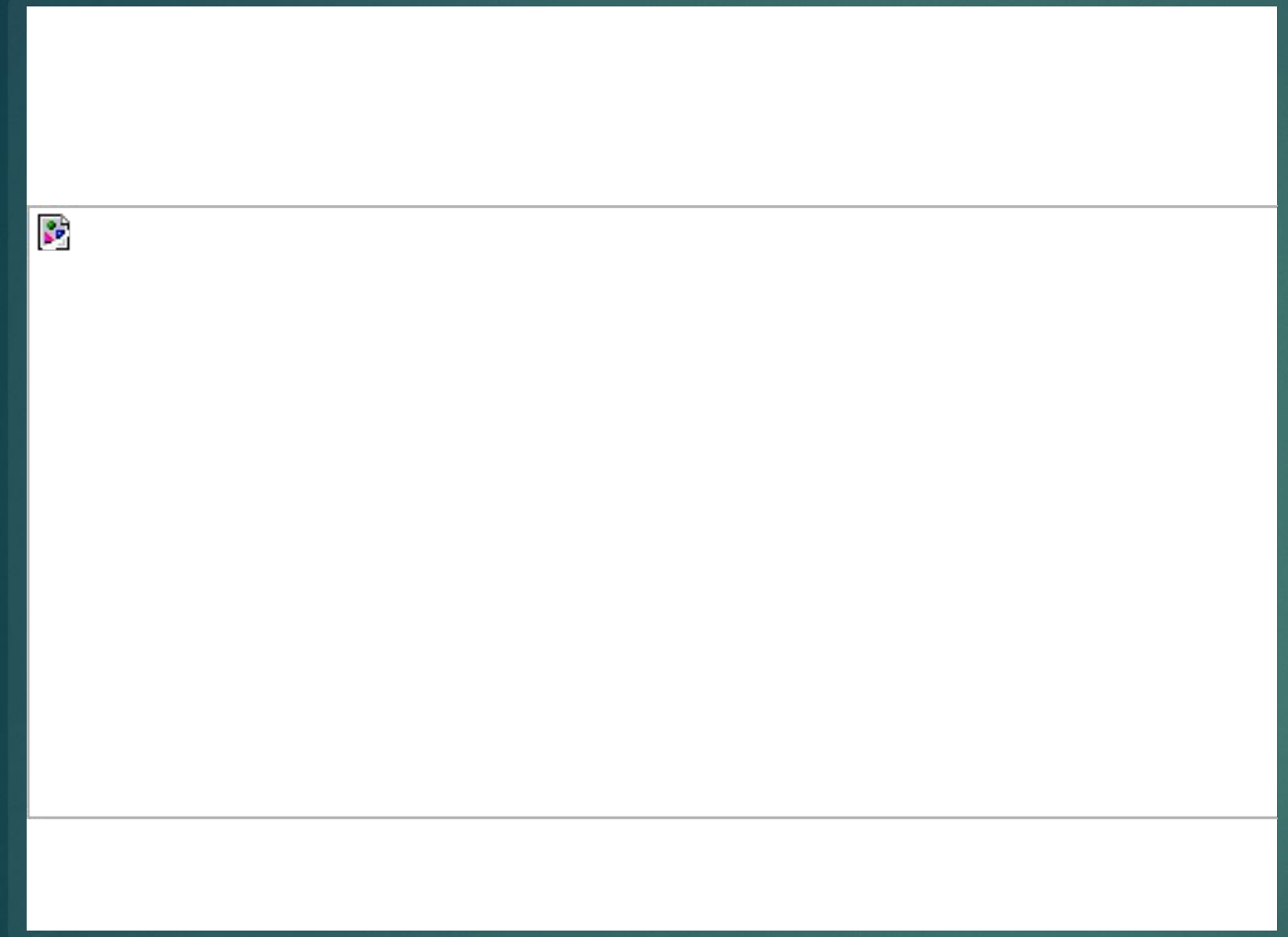
REFERENTIAL INTEGRITY, ASSERTIONS, TRIGGERS

Database Management System

SUJAN TAMRAKAR


Integrity Constraints
Domain Constraints
Table related constraints
Assertions
Check
Check
Column constraints
Table constraints
NOT NULL
Unique key
Primary key
Referential
Foreign key
Check
Unique key
Primary key
Check
Referential foreign
key



REFERENTIAL INTEGRITY

Referential Integrity

- Ensures that a value that appears in one relation for a given set of attributes also appears for a certain set of attributes in another relation.
- It states that table relationships must always be consistent.
- Foreign keys can be specified as part of the SQL create table statement by using the foreign key clause.
- The table containing the foreign key is called the child table, and the table containing the main (primary) key is called the referenced or parent table.
- By default a foreign key references the primary key attributes of the referenced table.
- Thus, any primary key field changes must be applied to all foreign keys, or not at all.

Referential Integrity

- The foreign key declaration specifies that for each Order tuple, the PersonId specified in the tuple must exist in the Persons relation.

```
CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)
);
```

“Persons" table:

“Orders” table:

PersonID	LastName	FirstName	Age

OrderID	OrderNumber	PersonID

Assertions

- An assertion is a predicate expressing a condition that we wish the db to always satisfy.
- Domain constraints, Referential integrity constraints are some examples of Assertions.

Syntax: create assertion <assertion_name> check <predicate>

- When an assertion is created, system tests it for validity. If assertion is valid, then any future modification to db is allowed only if it doesn't cause that assertion to be violated.
- This testing may introduce a significant amount of overhead if complex assertions have been made.

Assertions

The assertion is that not more than one valid contract per client is allowed.

```
CREATE ASSERTION asrt_OneValidContract CHECK (1 <= ALL(
SELECT COUNT(ClientID)
FROM Contract
WHERE getdate() BETWEEN ValidFrom and ISNULL(ValidTo, GETDATE())
GROUP BY ClientID))
GO
```

Other scenarios:
* sum of all loan amounts for each branch must be less than sum of all account balance at the branch.
* every loan has at least one customer who maintains an account with a minimum balance of $1000

Assertions

```
CREATE ASSERTION AT_MOST_ONE_PRESIDENT as CHECK
((select count(*)
    from EMP e
   where e.JOB = 'PRESIDENT') <= 1
);
```

This SQL statement creates an assertion to demand that there's no more than a single president among the employees.

```
CREATE ASSERTION NoRipoffs CHECK (
NOT EXISTS (
SELECT rest FROM Sells
GROUP BY rest
HAVING AVG(price) > 3
));
```

In Sells(rest, soda, price), no rest may charge an average of more than $3.

Triggers

- A statement that the system executes automatically as a side effect of a modification to the database.
- To design, following two requirements are needed:
  - Specify when a trigger is to be executed.
    This is broken up into an event that causes the trigger to be checked and a condition that must be satisfied for trigger execution to proceed.
  - Specify the actions to be taken when the trigger executes.
- Useful mechanisms for alerting humans or for starting certain tasks automatically when certain conditions are met.

Triggers

- Working:
  - The db stores triggers as if they were regular data so that they are persistent and are accessible to all db operations.
  - Once we enter a trigger into the db, the db system takes on the responsibility of executing it whenever the specified event occurs and the corresponding condition is specified.

Triggers

Scenario:

1. Instead of allowing negative account balances, the bank deals with overdrafts by setting the account balance to zero and creating a loan in the amount of the overdraft. The bank gives this loan a loan number identical to the account no. of the overdrawn account.
2. In Inventory System, when inventory level of an item falls below minimum set value, then an order should be placed automatically.
3. When applicant numbers are greater than available seats, reply them with some message (waiting preferably).

Triggers

Syntax:

```
CREATE TRIGGER trigger_name trigger_time trigger_event
ON table_name
FOR EACH ROW
BEGIN
...
END;
```

Example:

```
CREATE TRIGGER before_employee_update
    BEFORE UPDATE ON employees
    FOR EACH ROW
BEGIN
    UPDATE employees_audit
    SET action = 'update',
     employeeNumber = OLD.employeeNumber,
        lastname = OLD.lastname,
        changedat = NOW();
END
```

Triggers

Example:

```
CREATE TRIGGER overdraft_trigger AFTER UPDATE ON account
    REFERENCING NEW ROW AS nrow
    FOR EACH ROW
        WHEN NROW.BALANCE < 0
BEGIN ATOMIC
        INSERT INTO borrower (select customerName, accNo from depositor where
            nrow.accNo = depositor.accNo);
        INSERT INTO loan VALUES (nrow.accNo, nrow.branchName, -nrow.balance);
        UPDATE account SET balance = 0 where account.accNo = nrow.accNo;
END
```

Triggers

Example:

```
Create Trigger before_insert_studentage BEFORE INSERT ON student_age FOR EACH
ROW
BEGIN
IF NEW.age < 0 THEN SET NEW.age = 0;
END IF;
END
```

Now, the above trigger will automatically insert the age = 0 if someone try to insert age < 0.

Triggers

The triggering event and actions can take many forms:

- The triggering event can be insert or delete, instead of updates.
- For updates, the trigger can specify columns whose update causes the trigger to execute.
- The referencing old row as clause can be used to create a variable storing the old value of an updated or deleted row. The referencing new row as clause can be used with inserts in addition to updates.
- Triggers can be activated before the event instead of after the event to prevent invalid updates. (ex. To disallow loan scheme)

Thank You.