SQL - CONSTRAINTS

CIS-673

LECTURE#03

BY RAJ PATIL

2 OUTLINE

- Constraints
  - Primary Key
  - Not Null
  - Unique
  - Check
  - Foreign Key
- Default
- Cascade
  - Delete
  - Update

3 CONSTRAINTS

- Applied on columns
- More than one constraint can be applied on the same column
- Filters bad data, or the data that is not per requirement

4 EXAMPLE - WITHOUT CONSTRAINTS

```
CREATE TABLE employees (
        eid CHAR(3),
        ename VARCHAR(20),
        employer_name varchar(15),
        emailid VARCHAR(30),
        age INT,
        salary NUMERIC(9,2),
        address varchar(50)
);
```

5 PRIMARY KEY

- **Definition:** No duplicate and NULL values allowed in the column

- **Syntax:** ALTER TABLE <table_name> ADD CONSTRAINT <constraint_name> PRIMARY KEY <column_name >;

- **Example:** ALTER TABLE Employees ADD CONSTRAINT pk_employee PRIMARY KEY (eid);

6 NOT NULL

- **Definition:** No NULL values allowed in the column
- **Syntax:** ALTER TABLE <table_name> MODIFY <column_name> NOT NULL;
- **Example:** ALTER TABLE Employees MODIFY ename NOT NULL;

7 UNIQUE

- **Definition:** No duplicate allowed.
  - Same as Primary Key constraint, except allows null

- **Syntax:** ALTER TABLE <table_name> ADD CONSTRAINT <constraint_name> UNIQUE <column_name >;

- **Example:** ALTER TABLE Employees ADD CONSTRAINT unique_addr UNIQUE(Address);

8 CHECK

- **Definition:** Checks if the value in a column belongs to a specific range or list of values

- **Syntax:** ALTER TABLE <table_name> ADD CONSTRAINT <constraint_name> CHECK(range or list)

- **Examples:**
  - ALTER TABLE employees ADD CONSTRAINT chk_sal CHECK(salary>15000 and salary <1000000);

  - ALTER TABLE Employees ADD CONSTRAINT chk_status CHECK (Marital_Status **in** ('Single', 'Married', 'Divorced', 'Widowed', 'Unknown'));

9 DEFAULT

- **Definition:** Not a constraint. Instead, facilitates a value

- **Syntax:** ALTER TABLE <table_name> MODIFY (<column_name> default-value);

- **Example:**
  - ALTER TABLE Employees MODIFY (marital_status DEFAULT 'Unknown');
  - ALTER TABLE Employees MODIFY (address DEFAULT '123 main St, MI-49401');
  - insert into employees(eid, ename, salary, emailid, age, address) values(111, 'Sam', 70000, 'sam@gmail.com', 23, '123 main St');
  - select * from employees;

10 CONSTRAINTS – INSIDE CREATE STATEMENT (1)

- CREATE TABLE employees (

  eid CHAR(3) primary key,

  ename VARCHAR(20) not null,

  employer_name varchar(15),

  salary NUMERIC(9,2) CHECK(salary>15000 and salary <1000000),

  emailid VARCHAR(30) default('welcome@gmail.com'),

  age INT default(24),

  marital_status varchar(10) CHECK(Marital_Status in ('Single', 'Married', 'Divorced', 'Widowed', 'Unknown')),

  address varchar(50) unique);

CONSTRAINTS – INSIDE CREATE STATEMENT (II)

```
CREATE TABLE employees (
        eid CHAR(3),
        ename VARCHAR(20) not null,
        employer_name varchar(15) not null,
        salary NUMERIC(9,2),
        emailid VARCHAR(30) default('welcome@gmail.com'),
        age INT default(24),
        marital_status varchar(10),
        address varchar(50),
        constraint pk PRIMARY KEY(eid),
        constraint unique_addr UNIQUE(address),
        constraint chk_sal CHECK(salary>15000 and salary <1000000),
        constraint chk_status CHECK (Marital_Status in ('Single','Married','Divorced','Widowed','Unknown'))        );
```

12 FOREIGN KEY (REFERENTIAL INTEGRITY)

- **Definition:**
  - Foreign key column of child table references a primary key column of parent table.
  - Value being inserted in the Foreign key column (of the child table) should be one of the values from the primary key column (of the parent table).
  - Table with Primary key is called Parent Table.
  - Table with Foreign key is called Child Table.
  - The only constraint where 2 columns (from two different tables) are involved

13 PARENT TABLE - COMPANIES

- **Create Parent Table**, called 'Companies':
  - CREATE TABLE companies(

    company_name VARCHAR(15) primary key,

    address VARCHAR(50) not null,

    num_employees INT default 10

    );
  - INSERT INTO companies VALUES('Google', 'San Francisco, CA', 4500);
  - INSERT INTO companies VALUES('Amazon', 'Seattle, DC', 5000);
  - INSERT INTO companies VALUES('Facebook', 'New York, NY', 7000);
  - SELECT * FROM companies;

14 USING ALTER TABLE

- **Syntax:**
  - ALTER TABLE <child-table> ADD CONSTRAINT <constraint_name> FOREIGN KEY <column-name> REFERENCES <parent-table>(<column-name>)

- **Example:**
  - ALTER TABLE employees ADD CONSTRAINT fk_constraint FOREIGN KEY (employer_name) REFERENCES companies(company_name);

15 FK CONSTRAINT – INSIDE CREATE STATEMENT (1)

```
CREATE TABLE employees (
        eid CHAR(3) primary key,
        ename VARCHAR(20) not null,
        employer_name varchar(15) REFERENCES companies(company_name),
        salary NUMERIC(9,2) CHECK(salary>15000 and salary <1000000),
        emailid VARCHAR(30) default('welcome@gmail.com'),
        age INT default(24),
        marital_status varchar(10) CHECK(Marital_Status in ('Single', 'Married', 'Divorced', 'Widowed', 'Unknown')),
        address varchar(50) unique
);
```

16 FK CONSTRAINT – INSIDE CREATE STATEMENT (II)

- CREATE TABLE employees (

```
…            //column-info
…            //column-info
constraint pk PRIMARY KEY(eid),
constraint unique_addr UNIQUE(address),
constraint chk_sal CHECK(salary>15000 and salary <1000000),
constraint chk_status CHECK (Marital_Status in ('Single','Married','Divorced','Widowed','Unknown')),
constraint fk_constraint FOREIGN KEY (employer_name) REFERENCES companies(company_name)
);
```

17 INSERT - EXAMPLES

- **Example#1** //Successful
  - INSERT INTO employees VALUES('111', 'John', 'Google', 192032, 'john@gmail.com', 23, 'Single', '123 st, MI');
  - parent key (Google) is present in the parent table's company_name column

- **Example#2** //Unsuccessful
  - INSERT INTO employees VALUES('222', 'Dave', 'LinkedIn', 92032, 'dave@gmail.com', 32, 'Married', 'pacific st, MI');
  - Violation – parent key (Linkedin) not found in the parent table's company_name column

18 DELETE - CASCADE

- CREATE TABLE employees (

  eid CHAR(3) primary key,

  ename VARCHAR(20) not null,

  employer_name varchar(15) REFERENCES companies(company_name) on delete cascade,

  salary NUMERIC(9,2) CHECK(salary>15000 and salary <1000000),

  emailid VARCHAR(30) default('welcome@gmail.com'),

  age INT default(24),

  marital_status varchar(10) CHECK(Marital_Status in ('Single', 'Married', 'Divorced', 'Widowed', 'Unknown')),

  address varchar(50) unique);

- delete from companies where company_name = 'Google'; //record from employees working at 'Google' automatically get deleted
- select * from employees;

19 DELETE – SET NULL

- CREATE TABLE employees (

  eid CHAR(3) primary key,

  ename VARCHAR(20) not null,

  employer_name varchar(15) REFERENCES companies(company_name) on delete set null,

  salary NUMERIC(9,2) CHECK(salary>15000 and salary <1000000),

  emailid VARCHAR(30) default('welcome@gmail.com'),

  age INT default(24),

  marital_status varchar(10) CHECK(Marital_Status in ('Single', 'Married', 'Divorced', 'Widowed', 'Unknown')),

  address varchar(50) unique);

- delete from companies where company_name = 'Google'; //employer_name for records of employees working at 'Google' is set to 'Null'
- select * from employees;

20 CREATING AND DROPPING SEQUENCE

- Sequence
  - Creating sequence: parent table, before child table
  - Dropping sequence: child table, before parent table

- ON DELETE
- - on delete cascade: delete records from child, if deleted from parent
- - on delete set null: don't delete records from child, instead set them to null