Unit 5:

CONSTRAINTS AND DATA INTEGRITY

Using SQL Server

Topics of Discussion

- Relational Points of View
- Tables, Rows, and Columns
- Relations, Attributes, and Keys
- Constraints
- Indexes

“As a student, I would like to understand how data integrity and relations works so I can design an expansive, efficient database system.”

Relational Points of View

Three Groups

- ❑ Database-theoretical approach -think in terms of provability, mathematical set theory, propositional logic, focus on logical design (think game designers, engineers, etc.)
- ❑ "Just Build approach" – tend to think in terms of tables, rows, and columns, focus on physical design (think application programmers) -OLTP
- ❑ Flat file/storage approach – tend to think in terms of data warehouses, repositories, Data analytics - OLAP

Overview of Relational Databases

- Table
    - Matrix with columns and rows
- Columns
    - Represent different data fields
    - Characteristics or attributes about entity
- Rows
    - Contain individual records
    - Attributes about a specific instance of entity

Overview of Relational Databases (continued)

- Entity
  - Object about which you want to store data
  - Different tables store data about each different entity

- Relationships
  - Links that show how different records are related

Overview of Relational Databases (continued)

- Key fields
  - Main types of key fields

    - Primary
    - Candidate
    - Surrogate
    - Foreign
    - Composite

Primary Keys

- Column in relational database table whose value must be unique for each row
- Serves to identify individual occurrence of entity
- Every row must have a primary key
- Cannot be NULL
- NULL
- **Value is absent or unknown**
- **No entry is made for that data element**

Candidate Keys

- Any column that could be used as the primary key
- Should be a column that is unique for each record and does not change
- Example: SSN

Surrogate Keys

- Column created to be record’s primary key identifier
- Has no real relationship to row to which it is assigned other than to identify it uniquely
- Surrogate key values are automatically generated using an Identity property

Foreign Keys

- Column in table that is a primary key in another table
- Creates relationship between two tables
- Value must exist in table where it is the primary key

Composite Keys

- Unique key that is created by combining two or more columns
- Usually comprises fields that are primary keys in other tables

Constraints

- Rules that restrict data values that can be entered into column
- Types of constraints:
  - Integrity constraints (primary keys, foreign keys, composite keys)
  - Value constraints (Check, Not Null, Default, Unique, etc.)

Constraints

- Table constraint
  - Restricts data value with respect to all other values in table
- Column constraint
  - Limits value that can be placed in specific column
  - Irrespective of values that exist in other table rows
- Constraint definitions should be placed either:
  - At end of CREATE TABLE command after table columns declared
  - Within each column definition
  - New statement using ALTER TABLE command

Constraints

- Constraint naming convention
  - *`tablename_columnname_constraintid`*

Integrity Constraints

- Primary key
  - Syntax (within table definition)
    - `CONSTRAINT` *`constraint_name`* `PRIMARY KEY`
  - Syntax (at end of table definition)
    - `CONSTRAINT` *`constraint_name`* `PRIMARY KEY (`*`columnname`*`)`
  - Syntax (new command)
    - `Syntax`
    - `ALTER TABLE table_name`
    - `add CONSTRAINT constraint_name PRIMARY KEY (column1, column2, ...);`

Integrity Constraints

- Foreign key
  - Column constraint
  - Specifies that value user inserts in column must exist as primary key in referenced table
  - Syntax (placed at end of table definition)

```
CONSTRAINT constraint_name
FOREIGN KEY (columnname)
REFERENCES primary_key_tablename
  (primary_key_columnname)
```

Integrity Constraints

- Foreign key (continued)
  - Syntax (placed at end of table definition)

```
CONSTRAINT (columm) constraint_name REFERENCES
  primary_key_tablename (primary_key_columnname)
```

  - Syntax (new script)

```
ALTER TABLE table_name
  add CONSTRAINT constraint_name
  FOREIGN KEY (column1, column2, ... column_n)
  REFERENCES parent_table (column1, column2, ...);
```

- Composite key (Syntax)
  - `CONSTRAINT` *`constraint_name`* `PRIMARY KEY (`*`columnname1, columnname2`*` …)`

Integrity Constraints

- Value constraints
  - Column-level constraints
  - Restrict data values that users can enter
  - Commonly used value constraints
    - CHECK conditions
    - NOT NULL constraint
    - DEFAULT constraint
    - UNIQUE constraint

Check Constraints

ALTER TABLE table_name
add CONSTRAINT constraint_name CHECK (column_name condition)

- ALTER TABLE person
  add CONSTRAINT person_state_ck
    CHECK (state IN (‘OH’, ‘CA', ‘FL’));

- Possible check conditions:
  - CHECK (AGE BETWEEN 0 and 135)
  - CHECK (yesterday < getdate())
  - CHECK (state = “OH” and City = “Columbus”)
  - CHECK (begin < end)
  - etc…

Unique, Not Null, Default

- ALTER TABLE Person
  ADD CONSTRAINT person_SSN_cc UNIQUE (SSN);

- ALTER TABLE Person
  ADD CONSTRAINT CHECK person_Last_nn (lastName IS NOT NULL);

- ALTER TABLE Person
  ALTER COLUMN City SET DEFAULT ‘Columbus’;

Modifying Existing Column Data Definitions

- Modify existing column's data declaration
  - Syntax

```
ALTER tablename
  MODIFY(columnname new_data_declaration);
```

Deleting a Column

- Data stored in deleted column removed from database
- Syntax

```
ALTER TABLE tablename
DROP COLUMN columnname;
```

Renaming a Column

- Syntax

```
ALTER TABLE tablename
RENAME COLUMN old_columnname TO new_columnname;
```

Adding and Deleting Constraints

- Add constraint to existing table
  - Syntax

```
ALTER TABLE tablename
  ADD CONSTRAINT constraint_name constraint_definition;
```

- Remove existing constraint
  - Syntax

```
ALTER TABLE tablename
  DROP CONSTRAINT constraint_name;
```

Enabling and Disabling Constraints

- Constraint enabled
  - DBMS enforces constraint when users attempt to add new data to database
- Disable existing constraint syntax

```
ALTER TABLE tablename
DISABLE CONSTRAINT constraint_name;
```

- Enable existing constraint syntax

```
ALTER TABLE tablename
ENABLE CONSTRAINT constraint_name;
```

Indexes

- Used to increase performance usually for record retrieval
- Automatically creates an index for each UNIQUE or PRIMARY KEY declaration
- Be careful not to OVERUSE indexes
- B-Tree (OLTP)
- Bit Map (OLAP)

```
CREATE INDEX mytable_column_idx ON Person(LastName);
DROP INDEX index_name ON table_name;     --DROP index
```

Questions, comments, discussion