Database for Software Developers

Anis Uddin Ahmad

Data Integrity

Data Integrity

Data integrity is a concept and process that ensures the

- **accuracy,**
- **completeness,**
- **consistency**, and
- **validity**

of an organization's data.

Primary threats to Data Integrity

Data integrity may be compromised through...

- Human error (malicious or unintentional)
- Transfer errors (unintended alterations or data)
- Bugs, viruses/malware, hacking, and other cyber threats
- Compromised hardware, such as a device or disk crash
- Physical compromise to devices

Primary threats to Data Integrity

Data integrity may be compromised through...

- Human error (malicious or unintentional)
- Transfer errors (unintended alterations or data)
- Bugs, viruses/malware, hacking, and other cyber threats

- Compromised hardware, such as a device or disk crash
- Physical compromise to devices

Primary threats to Data Integrity

Data integrity may be compromised through...

- Human error (malicious or unintentional)
- Transfer errors (unintended alterations or data)
- Bugs, viruses/malware, hacking, and other cyber threats

➡ `Data validation, Encryption, Access control`

- Compromised hardware, such as a device or disk crash
- Physical compromise to devices

➡ `Data backup, duplication`

Types of Data Integrity

Data integrity may be compromised through...

- ☑ Entity Integrity
- ☑ Referential Integrity
- ☑ Domain Integrity
- ☑ User-Defined Integrity

Types of Data Integrity

Entity Integrity

Each row of a table has a unique and non-null primary key value.

id	name	email	password
204	Abdur Rahman	arahman@gmail.com	eyu2yg3i...
205	Abul Kalam	abulkalam234@gmail.com	3ewdyg3i...
206	Abdur Rahman	rahman.ctg@gmail.com	4ertgsv3i...
207	Muhammad Musa	md.musa.iiuc@gmail.com	jfue63hw...

Types of Data Integrity

Entity Integrity

Each row of a table has a unique and non-null primary key value.

id	name	email	password
204	Abdur Rahman	arahman@gmail.com	eyu2yg3i...
205	Abul Kalam	abulkalam234@gmail.com	3ewdyg3i...
206	Abdur Rahman	rahman.ctg@gmail.com	4ertgsv3i...
207	Muhammad Musa	md.musa.iiuc@gmail.com	jfue63hw...

Types of Data Integrity

Referential Integrity

If a value of one attribute refers to another (relation) attribute, then the referenced value must exist.

Types of Data Integrity

Referential Integrity

If a value of one attribute refers to another (relation) attribute, then the referenced value must exist.

id	name	email	password
204	Abdur Rahman	arahman@gmail.com	eyu2yg3i...
205	Abul Kalam	abulkalam234@gmail.com	3ewdyg3i...
206	Abdur Rahman	rahman.ctg@gmail.com	4ertgsv3i...
207	Muhammad Musa	md.musa.iiuc@gmail.com	jfue63hw...

users

id	user_id	date	checked_in
625	205	2023-11-22	2023-11-22 07:05:49
626	207	2023-11-22	2023-11-22 07:06:56
627	809	2023-11-22	2023-11-22 07:06:57

attendance

Types of Data Integrity

Referential Integrity

If a value of one attribute refers to another (relation) attribute, then the referenced value must exist.

id	name	email	password
204	Abdur Rahman	arahman@gmail.com	eyu2yg3i...
205	Abul Kalam	abulkalam234@gmail.com	3ewdyg3i...
206	Abdur Rahman	rahman.ctg@gmail.com	4ertgsv3i...
207	Muhammad Musa	md.musa.iiuc@gmail.com	jfue63hw...

users

FOREIGN KEY

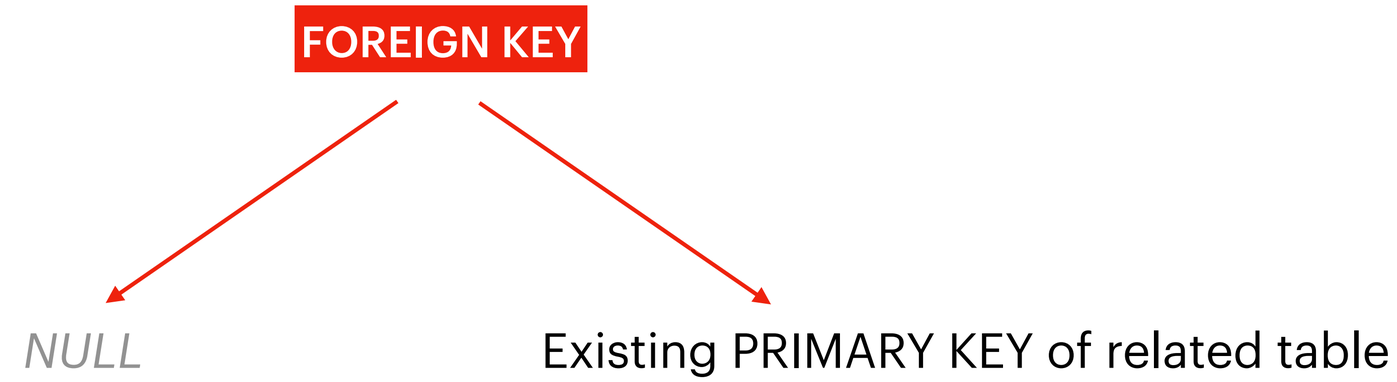
id	user_id	date	checked_in
625	205	2023-11-22	2023-11-22 07:05:49
626	207	2023-11-22	2023-11-22 07:06:56
627	809	2023-11-22	2023-11-22 07:06:57

attendance

Types of Data Integrity

Referential Integrity

If a value of one attribute refers to another (relation) attribute, then the referenced value must exist.



Types of Data Integrity

Domain Integrity

Defines a set of values or restriction on the values
that are acceptable (by domain) to be stored in a column.

Types of Data Integrity

Domain Integrity

Defines a set of values or restriction on the values that are acceptable (by domain) to be stored in a column.

- Driving license applicant age must be 16 or above
- In double entry accounting system, assets = liabilities + owners' equity.
- Total payment of an invoice cannot be negative
- No past date can be entered as the Expiry Date

Types of Data Integrity

Domain Integrity

Defines a set of values or restriction on the values
that are acceptable (by domain) to be stored in a column.

FOREIGN KEY

CHECK

NOT NULL

ENUM

Types of Data Integrity

User-defined Integrity

Defines a set of values, restrictions or format
that are acceptable (by user) to be stored in a column.

Types of Data Integrity

User-defined Integrity

Defines a set of values, restrictions or format that are acceptable (by user) to be stored in a column.

- Invoice number should be prefixed with "INV00"
- Student registration number format should be YYYY{6-DIGIT-SEQ}
- In a KYC Solution, value of *migrated* field for every new record is *false*

Types of Data Integrity

User-defined Integrity

Defines a set of values, restrictions or format that are acceptable (by user) to be stored in a column.

- Invoice number should be prefixed with "INV00"
- Student registration number format should be YYYY{6-DIGIT-SEQ}
- In a KYC Solution, value of *migrated* field for every new record is *false*

FOREIGN KEY CHECK NOT NULL *ENUM*

MySQL Constraints

and other features to ensure data integrity

- PRIMARY KEY
- CHECK
- ENUM
- FOREIGN KEY
- NOT NULL
- Transaction
- UNIQUE
- DEFAULT

Discussing today

In future classes, InSha-Allah

Discussed in earlier classes

MySQL Constraints

The UNIQUE Constraint

☑ Ensures that every value in a column is different

MySQL Constraints

The UNIQUE Constraint

- ☑ Ensures that every value in a column is different
- ☑ A table can have multiple UNIQUE column

MySQL Constraints

The UNIQUE Constraint

- ☑ Ensures that every value in a column is different
- ☑ A table can have multiple UNIQUE column

```
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    email VARCHAR(50) NOT NULL,
    password CHAR(32) NOT NULL
);
```

MySQL Constraints

The UNIQUE Constraint

- ☑ Ensures that every value in a column is different
- ☑ A table can have multiple UNIQUE column

```
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    email VARCHAR(50) NOT NULL,
    password CHAR(32) NOT NULL,
    UNIQUE (email)
);
```

MySQL Constraints

The UNIQUE Constraint

```
CREATE TABLE products (
    -- all the defined columns...

    UNIQUE (code)
);
```

MySQL Constraints

The UNIQUE Constraint

☑ Constraint can have a name (helpful for faster debugging)

```
CREATE TABLE products (
    -- all the defined columns...

    CONSTRAINT UNIQ_prod_code UNIQUE (code)
);
```

MySQL Constraints

The UNIQUE Constraint

- ☑ Constraint can have a name (helpful for faster debugging)
- ☑ May combine multiple columns

```
CREATE TABLE products (
    -- all the defined columns...

    CONSTRAINT UNIQ_prod_code UNIQUE (code),
    CONSTRAINT UNIQ_active_sku UNIQUE (sku, deleted_at)
);
```

MySQL Constraints

The UNIQUE Constraint

- ☑ Constraint can have a name (helpful for faster debugging)
- ☑ May combine multiple columns
- ☑ Can be defined after defining table by ALTERing

```
ALTER TABLE products ADD UNIQUE (sku);
ALTER TABLE products ADD CONSTRAINT UNIQ_sku UNIQUE (sku);
```

MySQL Constraints

The UNIQUE Constraint

- ☑ Constraint can have a name (helpful for faster debugging)
- ☑ May combine multiple columns
- ☑ Can be defined after defining table by ALTERing
- ☑ In fact, UNIQUE Constraint is an INDEX
- ☑ Can be removed if needed (as removing an INDEX)

```
ALTER TABLE products DROP INDEX UNIQ_sku;

-- To get a list of indexes
SHOW INDEXES FROM products;
```

MySQL Constraints

The CHECK Constraint

- Validates a value using an *expression*
- The *expression* must be evaluated to *TRUE* or *UNKNOWN*

MySQL Constraints

The CHECK Constraint

- ☑ Validates a value using an *expression*
- ☑ The *expression* must be evaluated to *TRUE* or *UNKNOWN*

```
CREATE TABLE manufacture_lots (
    id CHAR(10) PRIMARY KEY,
    product_id INT NOT NULL,
    manufacture_date DATETIME NOT NULL,
    expiry_date DATETIME,
    total_items INT,

    FOREIGN KEY (product_id) REFERENCES products(id),
    CHECK (expiry_date > manufacture_date)
);
```

MySQL Constraints

The CHECK Constraint

- ☑ Validates a value using an *expression*
- ☑ The *expression* must be evaluated to *TRUE* or *UNKNOWN*

```
CREATE TABLE manufacture_lots (
    id CHAR(10) PRIMARY KEY,
    product_id INT NOT NULL,
    manufacture_date DATETIME NOT NULL,
    expiry_date DATETIME,
    total_items INT,

    FOREIGN KEY (product_id) REFERENCES products(id),
    CHECK (expiry_date > manufacture_date)
);
```

MySQL Constraints

The CHECK Constraint

- ☑ Can be defined at Table or Column level
- ☑ Column level constraint can only use the current column

MySQL Constraints

The CHECK Constraint

- ☑ Can be defined at Table or Column level
- ☑ Column level constraint can only use the current column

```
CREATE TABLE manufacture_lots (
    id CHAR(10) PRIMARY KEY,
    product_id INT NOT NULL,
    manufacture_date DATETIME NOT NULL,
    expiry_date DATETIME,
    total_items INT CHECK (total_items > 0),

    FOREIGN KEY (product_id) REFERENCES products(id),
    CHECK (expiry_date > manufacture_date)
);
```

MySQL Constraints

The CHECK Constraint

- ☑ Can be used user defined names
- ☑ Database will generate a name if omitted, i.e. *{tbl}_chk_{seq}*

MySQL Constraints

The CHECK Constraint

- ☑ Can be used user defined names
- ☑ Database will generate a name if omitted, i.e. *{tbl}_chk_{seq}*

```
CREATE TABLE manufacture_lots (
    id CHAR(10) PRIMARY KEY,
    product_id INT NOT NULL,
    manufacture_date DATETIME NOT NULL,
    expiry_date DATETIME,
    total_items INT CONSTRAINT not_empty_lot CHECK (total_items > 0),

    FOREIGN KEY (product_id) REFERENCES products(id),
    CONSTRAINT future_man_date CHECK (manufacture_date > expiry_date)
);
```

MySQL Constraints

The CHECK Constraint

- Condition expressions must adhere to the following rules:
  - Column with AUTO_INCREMENT is not permitted
  - Literals, **deterministic** built-in functions, and operators are permitted
  - Stored functions, Stored procedure are not permitted
  - Variables are not permitted
    - System variables, user-defined variables, stored program local variables
  - Subqueries are not permitted

Example of **non-deterministic**: CONNECTION_ID(), CURRENT_USER(), NOW()

MySQL Constraints

The CHECK Constraint

☑ Can be added after table definition using ALTER TABLE

```
ALTER TABLE applicant_details
ADD CHECK (age ⩾ 16);
```

```
ALTER TABLE applicant_details
ADD CONSTRAINT DRIVER_MIN_AGE CHECK (age ⩾ 16);
```

MySQL Constraints

The CHECK Constraint

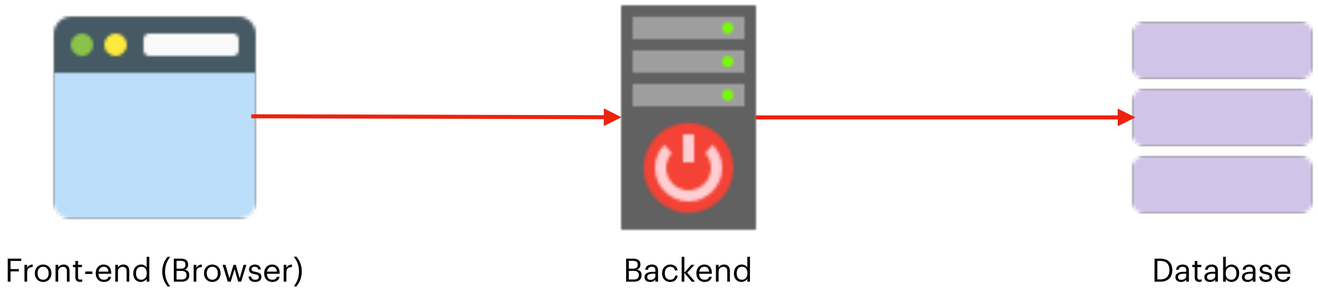
- ☑ Can be deleted using ALTER TABLE

```
ALTER TABLE manufacture_lots DROP CHECK future_man_date;

— Find all the CHECKs and their names
SHOW CREATE TABLE manufacture_lots;
```

Where to ensure integrity

Front-end, Backend or Database?



Questions?