PV168

Databases I

What is a Database

A database is an organized collection of structured information, or data, typically stored electronically in a computer system.

Multiple database types:

- Relational Databases
  - PostgreSQL, MariaDB, H2, Derby
- NoSQL Databases (non-relational)
  - MongoDB, Elasticsearch, Redis, DynamoDB
- Other

Why do we need a database

- store data (persistence, state, replication)
- structure (schema, relations)
- transactions (atomicity)
- data integrity (foreign keys, relationships, constraints)
- access control (locks, permissions)
- availability

JDBC – Java Database Connectivity

- API for communication with RDBMS
- Each DB has its own driver
  - Driver is a component that enables java application to interact with the database

JDBC URI

Examples:

- MySQL: `jdbc:mysql://server:port/dbname?characterEncoding=UTF-8`
- Postgres: `jdbc:postgresql://host:port/database`

H2 Database JDBC URI for specific modes:

- H2 (in-memory): `jdbc:h2:mem:database_name`
- H2 (embedded): `jdbc:h2:~/file/path/database.db;PROP=value`
- H2 (server): `jdbc:h2:tcp://localhost/~/db_name`

SQL - Structured Query Language

Types:

- a data query language (DQL) - `SELECT`
- a data definition language (DDL) - `CREATE, ALTER, DROP`
- a data control language (DCL) - access (`GRANT` , `DENY`)
- a data manipulation language (DML) - `INSERT` , `UPDATE` , `DELETE`

Note: Most of the DDL and DCL are implementation-specific (dependent on the RDBMS you are using)

Create a new Department Table

```
-- create a new Department table
CREATE TABLE IF NOT EXISTS Department
(
    id         BIGINT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    number     VARCHAR(10) NOT NULL UNIQUE,
    name       VARCHAR(50) NOT NULL,
    createdAt TIMESTAMP   NOT NULL DEFAULT CURRENT_TIMESTAMP
);
```

Create a new Employee Table

```
-- create a new Emplotee table
CREATE TABLE IF NOT EXISTS Employee
(
    id           BIGINT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    firstName    VARCHAR(150) NOT NULL,
    lastName     VARCHAR(150) NOT NULL,
    birthDate    DATE         NOT NULL,
    departmentId BIGINT REFERENCES Department (id),
    createdAt    TIMESTAMP    NOT NULL DEFAULT CURRENT_TIMESTAMP
);
```

Primary key

- Identification of objects (unique, immutable, not null)
- Should be synthetic
  - Identity (since SQL:2003):
    - `id BIGINT GENERATED ALWAYS AS IDENTITY PRIMARY KEY`
    - `id BIGINT GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY`
  - Seqence:

    ```
    CREATE SEQUENCE Department_Id_Seq AS BIGINT START WITH 1;
    ```

  - UUID

Relationships

Are used for the collections or composition representation

- One-to-one
- One-to-many
- Many-to-many

Foreign key

- A `FOREIGN KEY` is a field (or collection of fields) in one table, that refers to the `PRIMARY KEY` of another table.
- Examples:
  - `departmentId BIGINT REFERENCES Department (id),`
  - `FOREIGN KEY (Department_Id_FK) REFERENCES Department (id)`

Constraints and default indexes

Constraints:

- `UNIQUE` - column values must be unique
- `CONSTRAINT Employee_Name_UC UNIQUE (firstName, lastName)`

Index is automatically created for:

- primary key
- each unique constraints

Cascades

- Task 06: What happend when we tried to delete a department with employees
- What might help: `ON DELETE CASCADE`:
  - ```
    departmentId BIGINT REFERENCES Department (id) ON DELETE
    CASCADE,
    ```
- Cascades have problems
  - They might cause an outage 🤣
    ```

Information schema

- Bonus task
- Schema: `INFORMATION_SCHEMA`
- provides information about all of the tables, views, columns, procedures, indexes, relationships in a database
- Examples:
  - `SELECT * FROM INFORMATION_SCHEMA.INDEXES;`
  - `SELECT * FROM INFORMATION_SCHEMA.COLUMNS WHERE table_name = 'Department';`

Migrations

- way to manage database changes
- use existing tools for it
  - Java: Liquibase
  - Python: Alembic
  - GO: Migrate
- we will not be covering the migration on this course

Demo time

Now let's take a look at some examples and how to work with the IDEA's Console