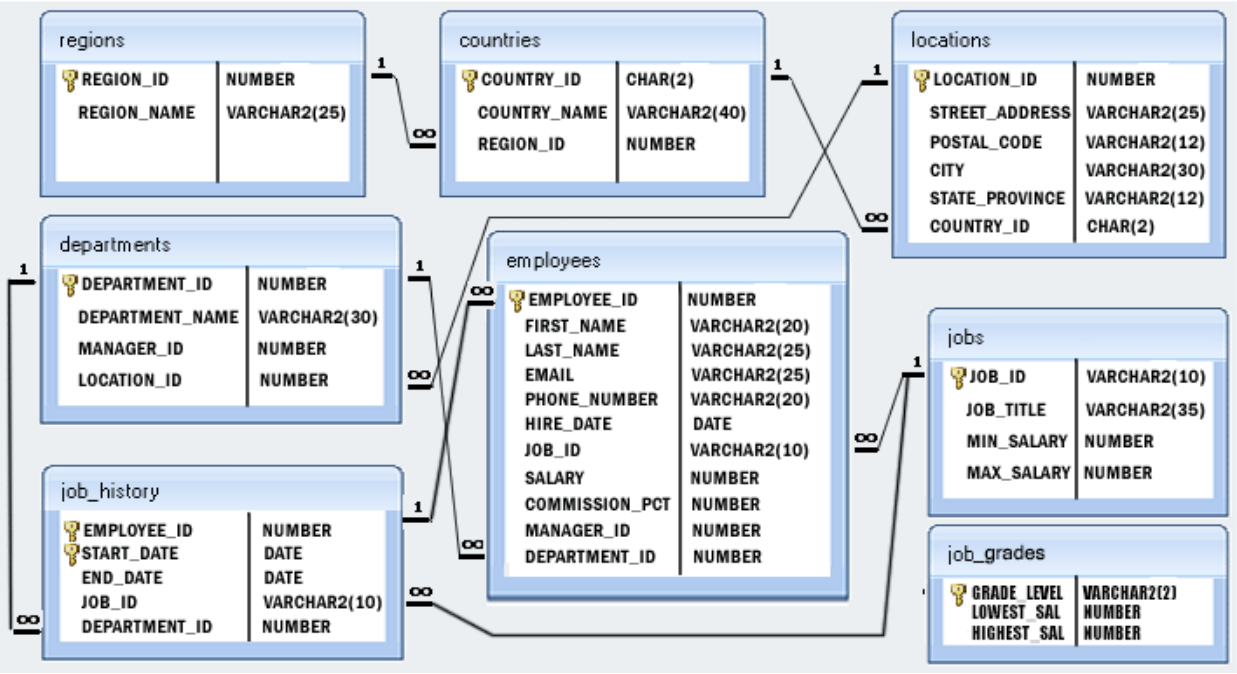
1. **Database sample: Human Resource (HR) Database**



2. **Database definition: Small guide**

- **Create/ drop table:**
  - Complex syntax, but do something simple

    References:

    https://docs.microsoft.com/en-us/sql/t-sql/statements/create-table-transact-sql?view=sql-server-ver15

    https://docs.microsoft.com/en-us/sql/t-sql/statements/drop-table-transact-sql?view=sql-server-ver15

  - Simple syntax:

```
CREATE TABLE
    { database_name.schema_name.table_name | schema_name.table_name
| table_name }
    ( { <column_definition> } [ ,...n ] )
[ ; ]
```

```
CREATE TABLE regions(
      region_id int NOT NULL,
      region_name varchar(25),
```

```
    CONSTRAINT pk_regions PRIMARY KEY (region_id) );
```

If you want *region_id* is auto increased from 1 with step 1 ➔ add **INDENTITY(1,1)** whe defining this column: `region_id int NOT NULL IDENTITY (1 , 1)`

```
DROP TABLE [ IF EXISTS ] { database_name.schema_name.table_name |
schema_name.table_name | table_name } [ ,...n ]
[ ; ]

DROP TABLE IF EXISTS regions;
DROP TABLE regions;
```

- Data types:

  https://docs.microsoft.com/en-us/sql/t-sql/data-types/data-types-transact-sql?view=sql-server-ver15

- **Alter table**: to add/modify/remove more columns, constraints
  - Create a new table without constraints:

```
CREATE TABLE countries(
    country_id int NOT NULL ,
    country_name varchar(40),
    region_id int
);
```

  - Add constraints: Add a primary key and a foreign key, check constraint:

```
ALTER TABLE countries ADD CONSTRAINT pk_countries PRIMARY KEY (country_id);

ALTER TABLE countries ADD CONSTRAINT fk_countries_2regions
    FOREIGN KEY (region_id) REFERENCES regions(region_id);

ALTER TABLE countries ADD CONSTRAINT chk_country_name
                                    CHECK (country_name <> 'ABC') ;
```

  - Remove constraints: Add a primary key and a foreign key:

```
ALTER TABLE countries DROP CONSTRAINT pk_countries ;
```

  - References: https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-table-transact-sql?view=sql-server-ver15

- Constraints: https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-table-column-constraint-transact-sql?view=sql-server-ver15

3. **Write SQL statement to define this database including all primary key and foreign key constraints**: (**please store all following statement in a SQL files named: studentName_studentID.sql)**

- ***Remark:*** Data types in the DB diagram may be not exists in SQL server => replace y the similar ones. For examples:

  NUMBER ➔ int or bigint

  Varchar2 ➔ varchar

1) Write a SQL statement to create a table **regions** including columns region_id, region_name and make sure that the column region_id will be unique and store an auto incremented value.
2) Write a SQL statements to create tables **countries** and **locations** including primary key and foreign keys
3) Write SQL statements to insert some records into these 3 tables. Ex.:

```
insert into regions(region_id, region_name) values(1,'Asia');
insert into countries(country_id, country_name, region_id) values(1,'Viet Nam', 1);
```

4) Write a SQL statement to create a table named **jobs** including columns job_id, job_title, min_salary and max_salary, and make sure that, the default value for job_title is blank and min_salary is 8000 and max_salary is NULL will be entered automatically at the time of insertion if no value assigned for the specified columns.
5) Write a SQL statement to alter table **jobs** to check whether the max_salary amount exceeding the upper limit 25000.
6) Write a SQL statement to create a table **departments** including columns department_id, department_name, manager_id, location_id and make sure that, the department_id column does not contain any duplicate value at the time of insertion, and the foreign key column location_id, refers to the column location_id of **locations** table, and another foreign key column manager_id, refers to the column

employee_id of **employees** table, can contain only those values which exist in the **employees** table.

7) Write a SQL statement to create a table **employees** including columns employee_id, first_name, last_name, email, phone_number hire_date, job_id, salary, commission, manager_id and department_id and make sure that, the employee_id column does not contain any duplicate value at the time of insertion, and the foreign key column department_id, refers to the column department_id of departments table, can contain only those values which exist in the departments table and another foreign key column job_id, refers to the column job_id of jobs table, can contain only those values which exist in the **jobs** table

8) Write a SQL statement to create a table **job_history** including columns employee_id, start_date, end_date, job_id and department_id and make sure that, the {employee_id, start_date} columns do not contain any duplicate values at the time of insertion and the foreign key column job_id contain only those values which exist in the jobs table.

9) Write SQL statements into insert sample records for each table that has been created.

10) Try to remove some records from countries and employees. Check if you can successfully remove them.

---

6 bis) Write a SQL statement to create a table **departments** including columns department_id, department_name, manager_id, location_id and make sure that, the {department_id, manager_id} columns do not contain any duplicate values at the time of insertion, and the foreign key column location_id, referenced by the column location_id of **locations** table, and another foreign key column manager_id, referenced by the column manager_id of **employees** table, can contain only those values which exist in the **employees** table.

7 bis) Write a SQL statement to create a table **employees** including columns employee_id, first_name, last_name, email, phone_number hire_date, job_id, salary, commission, manager_id and department_id and make sure that, the employee_id column does not contain any duplicate value at the time of insertion and the foreign key columns combined by *department_id and manager_id* columns contain only those unique combination values, which combinations exist in the **departments** table.
