HOME WORKS GUIDELINE. RELATIONAL DATABASE DESIGN.

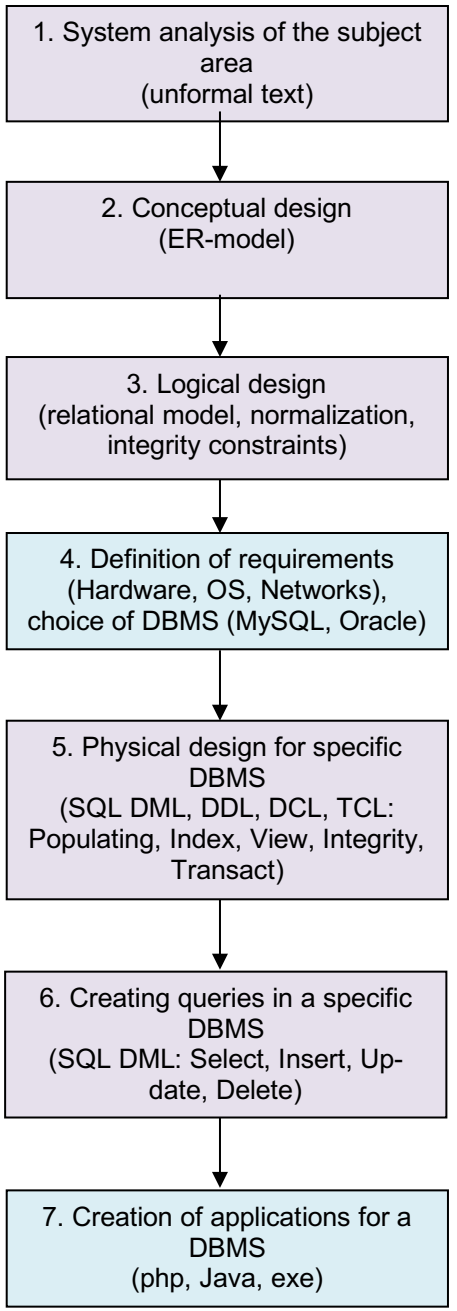
1. TARGET.

The purpose of a set of homework is to study methods and tools for designing relational databases.

The works variant number is generated from the student's last name and first name.

The task. For a certain subject area defined according to an individual variant, it is necessary to carry out conceptual, logical and physical database design, as well as create the required queries (SELECT) and representations (VIEW) for the database.

The stages of designing a relational database are shown in the figure to the right.



The task is performed sequentially in the form of several related homework (HW):

HW-01 covers the 1st and 2nd stages of database design, and executed after LW-01.
HW-02 covers the 3rd stage of database design, and executed after LW-02.
HW-03 covers part of the 5th stage of database design, and executed after LW-03.
HW-04 covers part of the 6th stage of database design, and executed after LW04-LW05.
*HW-05 covers part 5 and part 6 stages of database design, and executed after PW01-PW03.

The stages of the 4th and 7th database design are not presented in the works.

Work is performed under control of MySQL no lower than version 5.23.

A homework report is generated complex of HW01-HW05 as one doc(x) file.

2. Assignment.

2.0. Select Your individual variant Nr.

a) Write your LastName in English alphabet. Must be at least 6 letters, if not enough, then add the required number of letters from the FirstName (if not enough, then repeat LastName and FirstName).

For example, for student Yuriy Li there will be LIYURIY.

b) Replace the first 6 letters with their ordinal numbers in the alphabet, writing the numbers as two-digit decimal numbers.

For example, 12 09 25 21 18 09.

Aa 1	Bb 2	Cc 3	Dd 4	Ee 5	Ff 6	Gg 7
Hh 8	Ii 9	Jj 10	Kk 11	Ll 12	Mm 13	Nn 14
Oo 15	Pp 16	Qq 17	Rr 18	Ss 19	Tt 20	Uu 21
Vv 22	Ww 23	Xx 24	Yy 25	Zz 26		

c) Consistently add modulo 16 these 4 numbers and add 1

For example, (12+09+25+21+18+09)mod20 + 1 = 94mod20 + 1 = 14 +1 = 15.

d) The resulting will be your variant Nr.

For example, 15. The number 15 in the example has been replaced by 0.

You need to consistently perform a set of interconnected homework for solve one problem.

2.1. HW-01. Conceptual Database Design. Subject Area Analyze, ERD.

2.1.0. This work must be done after LW-01 (Creation ER Diagrams with Online CASE Tools).

2.1.1. To analyze the given Subject Area and the Prototype of the database model give a description of the thematic area, types of users, tasks to be solved, application goals, possible limitations and assumptions about the functioning of IS.

2.1.2. Identify and, if necessary, add the main and dependent entities.

2.1.3. Identify and, if necessary, add the simple and key attributes.

2.1.4. Build an entity-relationship diagram (ERD) in Crow's Foot notation using the visual databases design tools of the Draw.io (https://draw.io) or LucidChart (https://lucidchart.com).

2.1.5. Execute Self-check for Completeness and Correctness ERD.

2.2. HW-02. Logical Database Design. Data Dictionary, RDM, Normalization.

2.2.0. This work must be done after LW-02 (Mapping erd2rdm. Creation Relation Data Model with Online CASE Tools).

2.2.1. Transform ERD into a relational data model (RDM).

2.2.2. Check and if it is necessary to normalize the resulting RDM to 3NF or to BKNF.

2.2.3. Create a Data Dictionary (specify the composition, naming, data types, ranges and other integrity constraints for attributes).

2.2.4. Build RDM using dbDesigner database visual design tools (https://dbdesigner.net/).

2.2.5. Execute Self-check for Completeness and Correctness RDM.

2.3. HW-03. Physical Database Design. SQL DDL/DML: Create, Insert Design.

2.3.0. This work must be done after LW-03 (Creation MySQL Database on a XAMPP Stack or a Database Hosting).

2.3.1. Export RDM from dbDesigner to the target DBMS MySQL (**create.sql** script);

2.3.2. Import RDM into MySQL DBMS using **phpMyAdmin** or Adminer: debug the **create.sql** script (CREATE TABLE, ALTER TABLE);

2.3.3. Fill the database with a test data set (at least 5 rows for each relations): create and debug the **insert.sql** script (INSERT INTO).

2.4. HW-04. SQL DML: Select Design.

2.4.0. This work must be done after LW04-LW05 (SQL Exercises).

2.4.1. Make several selections of data from the created tables (write SELECT commands);

2.4.2. One of the requests should be written in two ways and explain which of the options will work faster and why.

2.5. Optional. HW-05. SQL DDL/DML: View, Update, Delete Design.

2.5.1. Create several views (CREATE VIEW);

2.5.2. For created views, it is necessary to check with the help of UPDATE, DELETE and INSERT queries whether they are updatable and explain the result.

3. Report.

3.1. Content of the Home Work Single Report.

A homework report is generated as a single docx file.

The report should include all of the components listed below in the form of text descriptions, tables, illustrations, hyperlinks:

1) Title page (name of student, group number, course name, homework name, variant number) (simple text).
2) The procedure for variant number generation and Task Assignment copy (simple text).

HW-01. Conceptual DB Design. Subject Area Analyze, ERD

3) Subject Area analysis. Document any assumptions and restrictions then you make on your Subject Area. (simple text)
4) Create Basic ERD on Chen's notation with draw.io site. (png and link)
5) Create Key Based ERD on Crow's Foot notation with LucidChart.com. (png and link)

HW-02. Logical DB Design. Data Dictionary, RDM, Normalization

6) Create a Data Dictionary Table (description of database tables, name and contents of the field, range of values, data type, integrity constraints: PK, FK, auto, unique, null, default, obligate). (simple table)
7) Build FA ERD = Basic RDM with DbDesigner.net (png, link)
8) Conclusion about Check of Normalization (1NF, 2NF, 3NF, BKNF). (simple text)

HW-03. Physical DB Design. SQL DDL/DML: Create, Insert Design.

9) The text of the debugged with phpMyAdmin script to create a database schema for MySQL (CREATE TABLE). (create.sql, simple text)
10) The text of the script to filling the MySQL database with test data (INSERT INTO VALUES). (insert.sql, simple text)

HW-04. SQL DML: Select Design.

11) Your five Queries to select data according to your task (SELECT FROM, SELECT FROM WHERE, SELECT FROM ORDER BY, SELECT COUNT FROM WHERE commands). (select-N.sql, simple text)

Optional. HW-05. SQL DDL/DML: View, Update, Delete Design.

12) Your three commands for creating data representations by assignment (CREATE VIEW) and commands for checking representations for updatability (UPDATE, DELETE, INSERT). (view-N.sql, simple text)

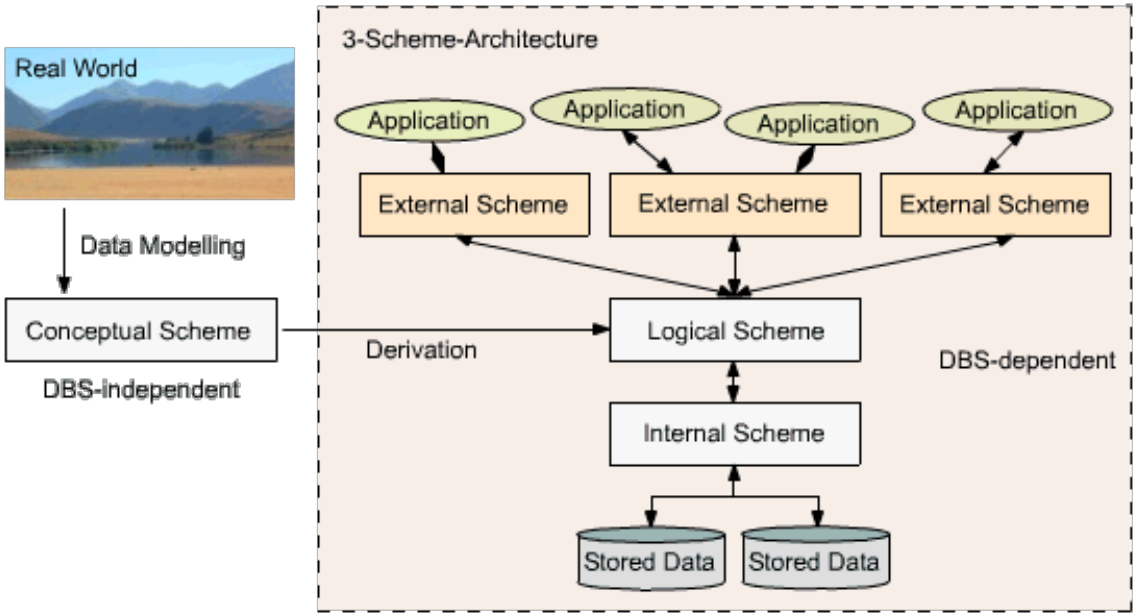
3.2. Graduation.

For a score of 10 points, you need to timely submit a report containing correctly and fully implemented paragraphs 2.1, 2.2, 2.3, 2.4.

3. Guideline.

3.1. Conceptual and Logical Design Self-check

3.1.1. Data Modelling Levels.



There are the following levels of data presentation - conceptual, logical, physical and application.

Database Development Life Cycle:

- Subject Area Analysis →
- Conceptual Model →
- Logical Design →
- Physical (Internal) Design →
- Application (External) Design →
- Optimize

1. The conceptual data model is an abstract view of data, at a conceptual level, data is represented as it looks in the real world, and can be named as it is called in the real world (for example, in Cyrillic and using special characters).

2. The logical data model is developed on the basis of a conceptual one for existing types of DBMS (for example, network, relational), but is in no way connected with a specific implementation of the DBMS and other physical conditions of implementation. It can be built on the basis of another logical model, for example, a data flow model or a process model.

3. The physical data model, on the other hand, depends on the specific DBMS being in fact a mapping of the system catalog. The physical model contains information about all objects in the database. Since there are no standards for database objects (for example, there is no standard for data types), the physical model depends on the specific implementation of the DBMS. Consequently, several different physical models can correspond to the same logical model.

4. The application (external) data model depends on the logical model and is a full or partial reflection (representation) of the logical data model for the end user; it can include validating the input data and formatting the output data. Usually there are many external views - different for each user.

3.2. Three Levels of Conceptual Model.

There are three levels of the conceptual model, which differ in the depth of presentation of information about the data:

1. Basic Entity Relationship Diagram (Basic ERD) is a top-level data model. It includes entities and relationships that reflect the core business rules of the domain. Such a diagram is not too detailed, it includes the main entities and relationships between them, which satisfy the basic requirements for an information system (IS). Basic ERD can include many-to-many relationships (not implemented in the DBMS) and not include key descriptions. Basic ERD is used for presentations and discussion of data structure with subject matter experts.

2. Key Based ERD (KB ERD) - A more detailed view of the data. It includes a description of all entities and primary keys and is intended to represent the data structure and keys that correspond to the subject area.

3. Fully Attributed ERD (FA ERD) - the most detailed representation of the data structure: includes all entities, attributes and relationships. Except for normalization, this model is almost identical to the Basic Relation Data Model (Basic RDM).

3.3. Self-check for Completeness and Correctness ERD.

1. The presence of the **goal and objectives** of the designed automated system.
2. Descriptions of the **Subject Area**: the presence of a description of roles, entities, attributes.
3. Availability of **Basic ERD**:
   a. The correct **set of entities**: completeness of coverage of the Subject Area, not overload (2 entities in 1), not redundancy (1 entity in 2).
   b. Correct **entity names** (syntax, semantics, not role).
   c. Correct **designation of entities** - primary, dependent-independent, associative (linking), characterizing (child).
   d. The correct **set of relationships**: designation, cardinality and obligatory.
4. Availability of **Key Based ERD**:
   a. The presence of **primary keys** - composite, surrogate; properties of uniqueness, static, not NULL; the domain of the PK values will remain constant over time.
5. Availability of **Full Attributive ERD**:
   a. The correct choice of a **set of attributes** (properties of information sufficiency and value; uniqueness of an attribute in a model; an attribute describes an instance, not a specific value; not polysemy).
   b. Correct choice of **attribute names** (you can use naming in the Object/Modifier/Class format (PersonFirstName, ProductSalesQuantity).

3.4. Self-check for Completeness and Correctness RDM.

1. Availability of **Basic RDM** (equivalent of Full Attributive ERD).
2. The correct **set of relations**.
3. The correct **set of relationships**.
4. The correct **set of attributes**.
    a. Absence of **repeating attributes**, for example: telephone-1, telephone-2 (they are taken out to another entity)
    b. Availability of the **minimum scope of the attribute** definition (the attribute must have at least 2 values, otherwise the attribute should be removed and transferred to the entity description).
    c. Availability for an **attribute parameters**: data type, mandatory mark, default value.
5. Correct choice of **primary keys**.
6. Correct choice of **foreign keys**.
7. Attribute **dependencies**:
    a. the values of the key attributes are independent of the values of any other attributes;
    b. unambiguous dependence of non-key attributes on the primary key;
    c. functional independence between non-key attributes, excluding derived (calculated) attributes (date of birth and age).
8. The presence of a **Normalized RMD**.
    a. The presence of **normalization**: placing an attribute in the appropriate entity to minimize anomalies when inserting, deleting, updating data and to minimize the use of null values for attributes.