Oracle SQL Data Modeler
Designs → Right button → New design

MAKE ER DIAGRAM
Logical → New entity → Draw entity form and fill attributes etc
Create relationship between entities

Creation of relational model (**table structure**)
Browser/Logical model → Right button → Engineer to Relational Model
✓ Logical, Entities, Relations → Engineer

Relational model to SQL code
"Generate DDL" button → Generate → "Drop" menu → ✓ Tables → OK → Copy/Save code

20 January 2023
Session 3, Exercises

SUPERMARKET

1. Identify entities and their attributes
Entities: tickets and products available
Attributes for **products**: name, brand, price, units in stock, barcode (5 numbers and a letter)
Attributes for **tickets**: total amount, date (day, time), from of payment (cash, card), box

2. Identify CK and select PK. Explain
Non-CK: name, brand, price, units of stock, total amount, form of payment
CK: barcode, date, box
PK: barcode, date and box

ARTIFICIAL INTELLIGENCE

1. Identify entities and their attributes
Entities: case and classification method
Attributes for **cases**: case ID, radiological measurements (average intensity of lesion, size, eccentricity and kurtosis of histogram), final diagnosis (inflammatory, ductal, carcinoma, lobular carcinoma or papillary carcinoma)
Attributes for **classification methods**: name, researchers who developed it, values of the parameters

2. Identify CK and select PK. Explain
Non-CK: radiological measurements, final diagnosis, values
CK: case ID, name, researchers
PK: case ID, name and researchers

```sql
INSERT INTO ticket
VALUES (3, '30-09-18', 'card', 3);

INSERT INTO ticket
VALUES (3.25, '01-10-18', 'card', 3);

INSERT INTO ticket
VALUES (3.4, '30-09-18', 'cash', 1);

INSERT INTO product
VALUES ('Milk', 'Vacona', 0.75, '56790A', 50);

INSERT INTO product
VALUES ('Milk', 'La munyidora', 1.25, '12345Z', 70);

INSERT INTO product
VALUES ('Beer', 'El monjo', 2.15, '54389Q', 100);

SELECT * FROM ticket;
SELECT * FROM product;
```

How to insert data into table: INSERT INTO —
VALUES (…);

How to print table: SELECT * FROM —

NAME	BRAND	PRICE	BARCODE	UNITS
Milk	Vacona	0,75	56790A	50
Milk	La munyidora	1,25	12345Z	70
Beer	El monjo	2,15	54389Q	100

Theory exam example

4. Define the following concepts

Change propagation refers to the process of automatically updating data in multiple locations within a DBMS when changes are made to one set of data. There are several different rules for change propagation:

- **Cascading Updates**: when a change is made to a PK value in one table, the corresponding FK values in all related tables are automatically updated to reflect the new value. This helps to ensure that all related data remains consistent and up-to-date.
- **Cascading Deletes**: when a row is deleted from a table, all related rows in other tables are also deleted automatically. This helps to ensure that there are no abandoned rows in the database.
- **Nullifying FK References**: when a row is deleted from a table, any FK references to that row in other tables are set to *null*. This helps to avoid referential integrity violations.
- **Restricted Updates**: if a change is made to a PK value in one table that would affect a FK value in another table, the update is restricted and an *error* is thrown. This helps to ensure that related data remains consistent and avoids data corruption.
- **Restricted Deletes**: if a row is attempted to be deleted from a table that has related rows in other tables, the delete is restricted and an *error* is thrown. This helps to avoid abandoned rows in the database and ensures referential integrity.

5. Define the following concepts

Primary key: candidate key (minimum set of attributes that uniquely identify each instance) chosen by the designer. A PK has two important properties:

  - Uniqueness: every value in the PK field must be unique. No two rows can have the same value in the PK field
  - Non-nulability: the PK field cannot be NULL. Every row in the table must have a value in the PK field

Discriminant: set of attributes which allow to distinguish between instances of the weak entity that depend on the same instance of the strong entity. For example, if we have a DB that tracks employees and contractors, we might use a discriminator attribute to differentiate between the two types of workers.

Simple attribute: attribute that **cannot** be further subdivided into smaller components. For example, if we have a DB of customers, a simple attribute might be the customer's *name* or *phone number*. These attributes are atomic and cannot be broken down any further

Composite attributes: attribute can be further divided into smaller components. For example, if we have a DB of *addresses*, a composite attribute might be the address, which can be divided into components as *street name*, *city*, *state* and *postal code*.

Single-valued attributes: attributes that have only one value for a given entity. For example, the *age* attribute of a person is single-valued, since each person has only one age.

Multivalued attributes: attributes that can have multiple values for a given entity. For example, the *skills* attribute of an employee might be multivalued, since an employee could have multiple skills, such as *programming*, *writing* and *public speaking*.

6. Explain, briefly, the ANSI-SPARC architecture: its components and the main properties

ANSI-SPARC architecture is a framework for designing and building DBMSs that consists of three main components: the external level, the conceptual level and the internal level.

- **External level:** this is the user interface to the DB system, where the user interacts with the DBMS through a set of user views/schemas. The EL represents the way users perceive the data.
- **Conceptual level:** this level provides a conceptual or abstract view of the entire DB system, which is idependent of any particular user view or application program. The CL specifies the structure of the data, the relationships among the data elements and the constraints on the data.
- **Internal level:** deals with the physical storage devices. The IL defines the physical representation of the data and the algorithms used to access and manipulate the data.

The main properties of the ANSI-SPARC architecture include:

- **Data independence**: provides a clear separation between the different levels, allowing for changes to be made at one level without affecting the others. This provides ↑ degree of the data independence, which mantains the integrity and consistency of data
- **Abstraction**: makes easier to understand and use the DBMS. The CL provides a conceptual view of the data, while the external level provides a user-friendly interface that is adjusted to the needs of the users.
- **Modularity**: each level can be designed and implemented independently of the other levels.
- **Flexibility**: can accommodate different types of data and different types of applications. This makes it suitable for a wide range of uses, from small personal DBs to large enterprise-level systems.

7. Enumerate and briefly describe the phases of design of a DB

The design of a DB involves several phases, which are often organized into a design methodology.

1. **Requirements analysis:** understand the user requirements, data needs and rules that the DB should satisfy. It includes identifying data entities, attributes and relationships, as well as the operations that users will perform on the data (transactions)
2. **Conceptual design:** the ↑-level design of the DB is created, which includes the conceptual data model. The CDM is a representation of the data and relationships between data entities without any implementation details. The DM is usually expressed in and Entity-Relationship (ER) diagram.
3. **Logical design:** translating the CDM into a logical data model. The LDM includes more details than the CM, such as PKs, FKs and data types. It is usually expressed in a DDL, such as SQL.
4. **Physical design:** the LDM is mapped to a physical data model that specifies how the data will be stored in the DBMS. The physical design includes the physical schema, indexes and other storage structures.
5. **Implementation:** the DB is created and populated with data. The logical and physical design is implemented using a DBMS and a specific DB model
6. **Testing and evaluation:** the DB is tested and evaluated to ensure that is meets the user requirements and is performing well. This includes testing the data integrity, security and performance of the DB
7. **Maintenance and evolution:** maintaining and evolving the DB as new requirements arise or as the data changes. This includes monitoring the performance of the DB, fixing issues and updating the design to accomodate new requirements or data