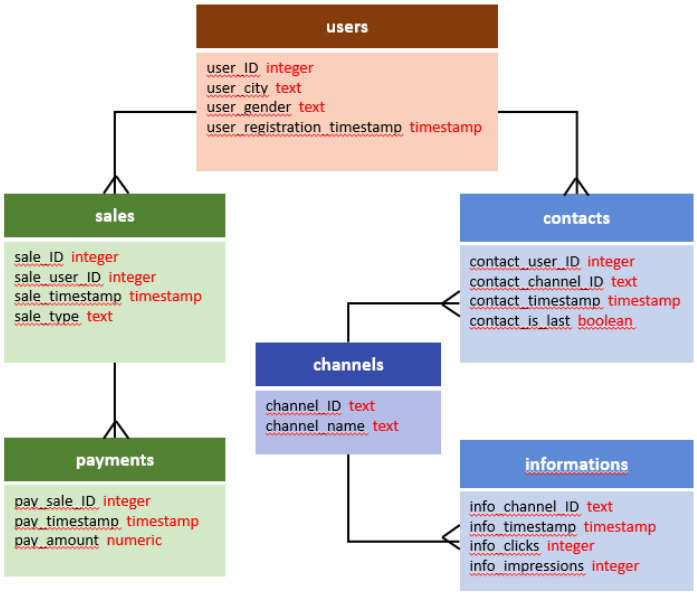
BI Specialist assignment - SQL and Data Literacy

Please see on the right a DWH diagram of an e-commerce platform.

In the **`informations`** table you can find the sum of all online marketing advertisement impressions (**`info_impressions`**) and clicks (**`info_clicks`**) for the respective event timestamps and the respective marketing channels, through which the (potential) customers came. The table does not contain details about individual customers.

The **`channels`** table is a lookup between channel names like "SEM" or "AFF" and the assigned alphanumerical channel IDs.



In order to perform activities other than advertisement views and clicks, a customer (user) needs to register on the platform, meaning to provide some personal details like address or gender. Not all of these details are mandatory for the registration. At the moment the customer registers on the platform, a row is created in the **`users`** table and a user ID is assigned to the customer.

All online advertisement views and clicks during a customer's lifecycle are tracked (if possible), before as well as after registration. The aggregates of these events are provided in the **`informations`** table, see above.
Additionally, if applicable, at the moment of registration rows are created in the **`contacts`** table for all of the customer's previous advertisement clicks with the respective user ID, channel and timestamp of the click (**`contact_timestamp`** = **`info_timestamp`**).
After the registration any tracked advertisement click by the customer is also stored in the **`contacts`** table. The registration itself is not considered a contact. The **`contact_is_last`** field is always updated accordingly and at any point time it is true for only one row per customer.

When the registered customer does a purchase on the platform, it creates a row in the **`sales`** table and when the item is paid, a row in the **`payments`** table.

The relations between tables are defined by the names, e. g. **`contacts.contact_user_ID`** refers to **`users.user_ID`**. A registered user does not need to have any contacts or sales.

For each task below (1a to 3b) only one query is expected, which would give exactly one result set as output. These result sets should contain all (and only) the dimensions and aggregations mentioned in the respective tasks. The tasks 1b and 3b also require further written explanations.

Please do not create a calendar for any of the tasks, but rather use only the provided time fields in the tables. Days without results should be skipped.

For the SQL queries, please choose any SQL dialect you prefer.

1) Please write one SQL SELECT query for each of the described results and answer the questions in b).
   a) The number of users per channel and registration date
   b) What could happen, if the key word DISTINCT is omitted in this query?
      Why does the correct number of users not match with the sum of users in this result, even if every user had a contact, and how would you adjust the query to fix it?

2) Please write one SQL SELECT query for each of the following results:
   a) The total **`pay_amount`** per city for sales in 2020
   b) The number of users, for whom the last contact has **`channel_name`** "SEM",
      but who have also at least one additional contact with another channel
   c) Per registration date and last channel, the number of users and the number of sales done by users within 3 days after the registration

3) Please write one SQL SELECT query for each of the following results:
   a) The number of clicks and registrations per day in 2020
   b) The number of clicks and registrations per click day in 2020
      Explain in writing: What would be measured with a) and what with b)?

Please note that no physical database will be provided for the above tasks. This is due to the fact, that any possible data set should be considered and not only a certain data sample, which is why we strongly recommend not to create a sample database for testing. In case of huge databases, special cases are quickly overlooked when testing queries only on samples.