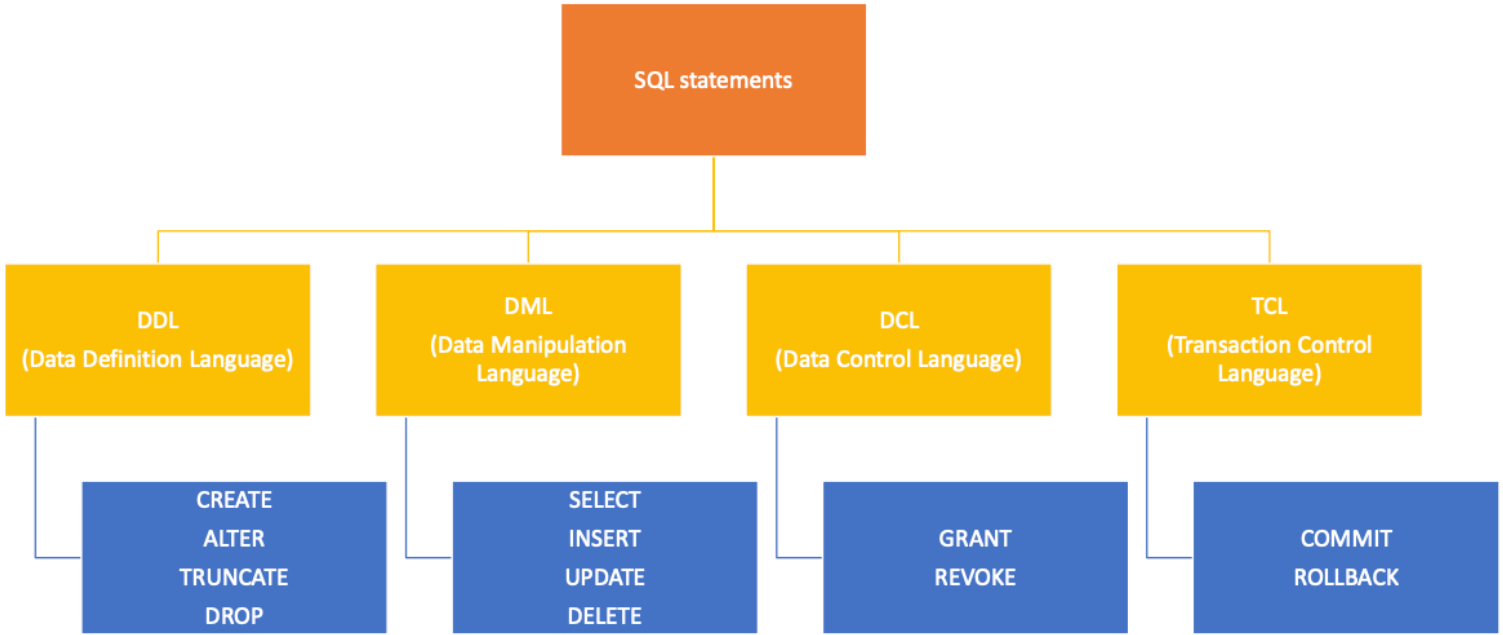
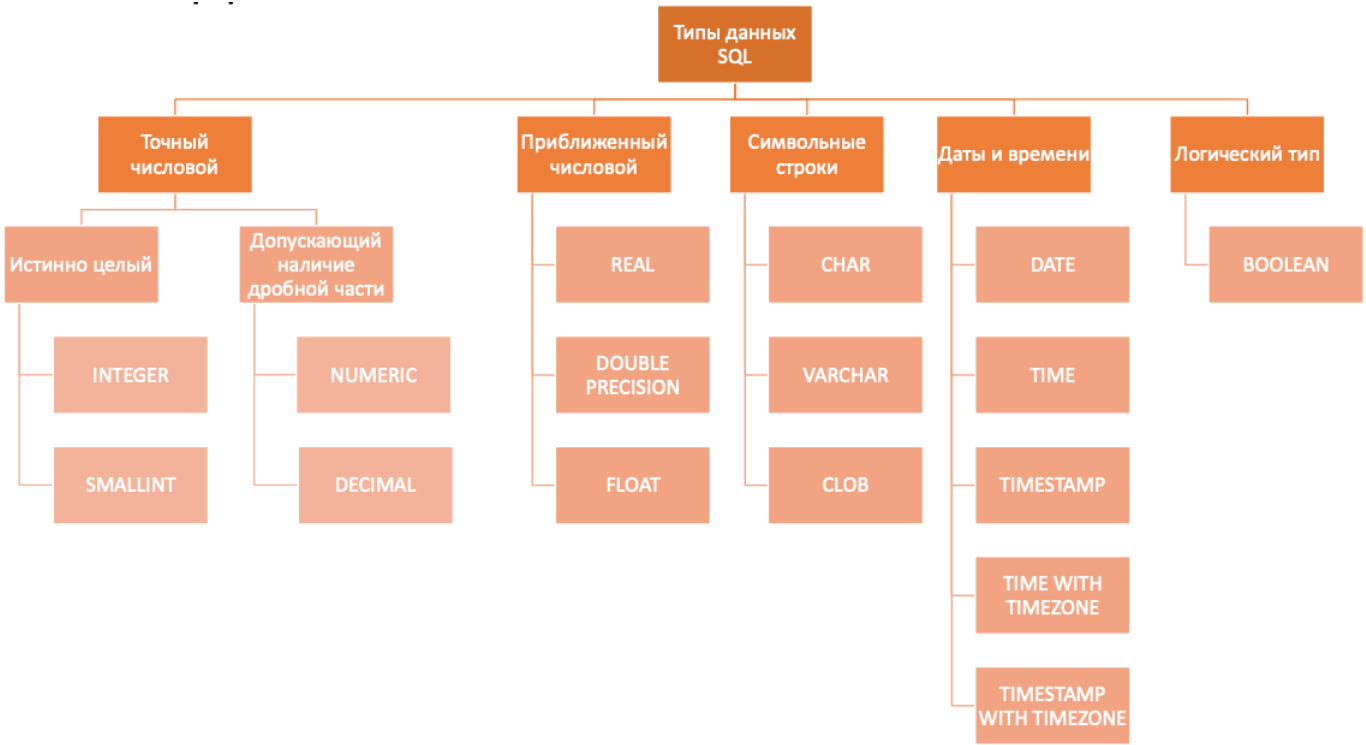
Databases

Seminar 1

SQL queries


SQL statements
DDL
(Data Definition Language)
DML
(Data Manipulation Language)
DCL
(Data Control Language)
TCL
(Transaction Control Language)
CREATE
ALTER
TRUNCATE
DROP
SELECT
INSERT
UPDATE
DELETE
GRANT
REVOKE
COMMIT
ROLLBACK

SQL datatypes



Task 1

Rank	Team	Nation	Total Points	Men	Women	Pairs	Ice Dance
1	ROC	ROC	74	17	20	19	18
2	United States of America	USA	65	18	13	14	20
3	Japan	JPN	63	19	18	16	10
4	Canada	CAN	53	9	16	13	15
5	People's Republic of China	CHN	50	12	7	18	13

Query structure

SELECT [DISTINCT] select_item_comma_list -- list of answer columns
FROM table_reference_comma_list -- list of tables
[WHERE conditional_expression] -- conditions for filtration, one can use AND / OR / NOT
[GROUP BY column_name_comma_list] -- grouping condition
[HAVING conditional_expression] -- conditions for filtration after the grouping
[ORDER BY order_item_comma_list]; -- list of fields for sorting

Order of execution

FROM → WHERE → GROUP BY → HAVING → SELECT → ORDER BY

Aggregation functions

When grouping, the SELECT block may contain either attributes by which grouping occurs, or attributes that are supplied as input to aggregating functions. There are 5 standard aggregation functions in SQL. When executing a function call, the special value NULL, which denotes a missing value, is not taken into account.

Aggregation functions

count() – number of records with a known value. If you need to count the number of unique values, you can use count(DISTINCT field_nm)

max() - the largest of all selected field values

min() - the smallest of all selected field values

sum() - the sum of all selected field values

avg() - average of all selected field values

Special functions

IN - belonging to a specific set of values:
X IN (a1, a2, ..., an) ≣ X = a1 or X = a2 or ... or X = an

BETWEEN - belonging to a certain range of values:
X BETWEEN A AND B ≣ (X >= A and X <= B) or (X <= A and X >= B)

LIKE - text satisfaction to the pattern: X LIKE '0%abc_0', where _ is exactly 1 character, and % is any sequence of characters (including zero length).

SIMILAR TO - matching text to SQL regular expression (POSIX-like):
'abc' SIMILAR TO '%(b|d)%'

Many conditions

```
SELECT
    IF number = 0 THEN
        'zero'
    ELSIF number > 0 THEN
        'positive'
    ELSIF number < 0 THEN
        'negative'
    ELSE
        'NULL'
    END IF AS number_class
FROM
    numbers
```

Many cases

```
SELECT
    CASE
        WHEN number = 0 THEN
            'zero'
        WHEN number > 0 THEN
            'positive'
        WHEN number < 0 THEN
            'negative'
        ELSE
            'NULL'
    END CASE AS number_class
FROM
    numbers
```

Unique values

```
SELECT
    count(DISTINCT ON department_nm)
FROM
    salary;
```

Dates in PostgreSQL

data type	description	example	output
TIMESTAMP	date and time	TIMESTAMP '2023-04-10 10:39:37'	2023-04-10T10:39:37
DATE	date (no time)	DATE '2023-04-10 10:39:37'	2023-04-10
TIME	time (no day)	TIME '2023-04-10 10:39:37'	10:39:37
INTERVAL	interval between two date/times	TIME '2023-04-10 10:39:37'	1 day, 2:00:10

Dates in PostgreSQL

Today's date can be found like this: NOW() (timestamp), CURRENT_DATE (date), LOCALTIME (current time)

DATE_TRUNC(‘[interval]’, time_column) - allows you to round to month, year, date, etc.
Example: DATE_TRUNC('month', DATE '2023-04-10') will return DATE '2023-04-01'
DATE_TRUNC('day', TIMESTAMP '2023-04-10 10:39:37') will return TIMESTAMP '2023-04-10 00:00:00'

TO_CHAR([date type], [pattern]) - allows you to format the date into a string using a pattern
Example: TO_CHAR(DATE '2023-04-10', 'YY') will output the last 2 digits of the year 23