Оглавление

Total

SQL для начинающих: с нуля до сертификата Oracle (Заур Трегулов)
https://coursehunter.net/course/sql-dlya-nachinayushchih-s-nulya-do-sertifikata-oracle

DML = SELECT, INSERT, UPDATE, DELETE, MERGE
TCL = COMMIT, ROLLBACK, SAVEPOINT
DDL = CREATE, ALTER, DROP, RENAME, TRUNCATE
DCL = GRANT, REVOKE

```
        //working in dual table (for testing operations)
select * from dual;

        //get date format from session_parameters
select * from nls_session_parameters where parameter='NLS_DATE_FORMAT';
alter session set NLS_DATE_FORMAT = 'DD.MM.RRRR HH24:MI:SS';
select * from v$version;
```

user = username, password
schema = has objects with uniq name

table_name = 1-30 char, first letter

```
        //get info about table
DESCRIBE schema.table_name;
```

user SYS = data dictionaries
user SYSTEM = admin, monitoring

```
        //get all objects
select * from dba_objects;
select * from all_objects; - can use
select * from user_objects;

select object_type, count(1)
from dba_objects
where owner = 'EDATA_PORTAL'
group by object_type
order by object_type
;

        //get tables
select table_name from dba_tables where owner = 'EDATA_PORTAL';
select column_name, data_type, nullable from dba_tab_columns;
```

//connect to other data base from this data base

нужны привилегии на работу с линками

1.

```
CREATE DATABASE LINK PRODLINK
CONNECT TO vnovikov
IDENTIFIED BY QAautomation
USING '(DESCRIPTION = (ADDRESS_LIST = (ADDRESS =
(PROTOCOL=TCP)(HOST=10.201.0.143)(PORT=1521))) (CONNECT_DATA=
(SID=<liferay>)))';
```

2.

```
SELECT * FROM EDATA_PORTAL.EDATA_DICT_KVK@PRODLINK;
```

//**вывести N записей**

```
select *
from WHM.viw_ob_inco_mb
where rownum <= 5
--fetch first 10 rows only
;
```

Описание БД

Шаги при создании ERD
• определить сущности;
• определить атрибуты сущностей;
• определить первичные ключи;
• определить отношения между сущностями;
• определить кардинальность;
• нарисовать ERD;
• проверить ERD
	Модель данных
	• определяет абстракцию данных для приложений;
	• включает:
	  • структуры данных;
	  • операции;
	  • зависимости;
	  • ограничения.

Объекты БД • таблицы (TABLE) • индексы (INDEX) • правила целостности (CONSTRAINT) • представления (VIEW) • процедуры (PROCEDURE) • функции (FUNCTION) • триггеры (TRIGGER)	Концептуальные модели данных • иерархическая; • сетевая; • реляционная; • объектно-реляционная.

CASE

```
SELECT field1, field2, salary,
CASE  salary
      WHEN min_salary THEN 'min'
      WHEN max_salary THEN 'max'
      ELSE 'medium'
END AS salary_range
FROM table1
ORDER BY
CASE salary_range
    WHEN 'max' THEN 1
    WHEN 'min' THEN 2
    ELSE 3
END
;
```

```
select distinct code_group
from
(
select COD_CONS_FK,
case
when  COD_CONS_FK  like('0%')  then
'begin'
when  COD_CONS_FK  like('1%')  then
'total'
else 'qwerty'
end as code_group
from VIW_OB_CONS_FK
)
;
```

```
WITH table1 AS(
  SELECT name, COUNT(*) cnt
  FROM table2
  JOIN table3
  ON table2.field = table3.field
)
SELECT name, cnt
FROM table1
WHERE cnt = (
  SELECT MAX(cnt)
  FROM table1
)
;
```

```
SELECT companyId, salary,
LISTAGG(name, ';')
WITHIN GROUP(
  ORDER BY salary DESC
) AS list
FROM client
GROUP BY companyId, salary
;
```

https://coursehunter.net/course/sqlfortesters

Имя БД select user||'@'||global_name from global_name;	Все таблицы БД select * from user_objects; select * from all_users; select * from all_tables; select * from all_tab_columns; select * from all_views;
Таблица из другой БД schema_name.Table_name@DB_link;	Свойства полей таблицы DESCRIBE <tableName> DESCRIBE user_source DESCRIBE trigget_source

Просмотр ограничений

SELECT * FROM USER_CONSTRAINTS
SELECT * FROM ALL_CONSTRAINTS

Типы ограничений

P - primary key - первичный ключ
R - reference - foreign key - внешний ключ
C - check - проверка данных
U - unique - все значения уникальны

```
SET SERVEROUTPUT ON

SELECT COUNT(1)
FROM ALL_TAB_COLUMNS
WHERE OWNER='WHM'
        AND TABLE_NAME = 'VIW_OB_CONS_MB';
```

//вывести 5 или 10 записей

```
select *
from WHM.viw_ob_inco_mb
where rownum <= 5
--fetch first 10 rows only
;
```

Несколько таблиц

Работа со столбцами

JOIN - внутреннее соединение
LEFT JOIN - левое внешнее соединение
RIGHT JOIN - правое внешнее соединение
FULL JOIN - полное внешнее соединение
CROSS JOIN - декартово произведение
SELECT * FROM table1 JOIN table2 ON table1.field = table2.field;

Работа со строками

В запросах должны совпадать = число колонок, верхнеуровневый тип данных.
Нельзя использовать вместе с колонками типа LOB, LONG, FILE.

```
SELECT COUNT(*)
FROM
(
   SELECT DISTINCT SRC_CD
   FROM WHM.VIW_OB_DOV_MB
   UNION
   SELECT DISTINCT COD_BUDGET
   FROM WHM.VIW_OB_INCO_MB
)
;
```

ALL - без учета уникальности значений
UNION - объединение множеств уникальных значений (DISTINCT)
INTERSECT - пересечение множеств уникальных значений (DISTINCT)
EXCEPT / MINUS - разность множеств уникальных значений (DISTINCT)

Одна таблица

NVL - заменяет NULL на переданное значение = NVL(fieldName, value)
сортировка = select * from VIW_OB_CONS_FK **order by** PLAN_BEGIN_YEAR_AMT **nulls last**; (first)

SYSDATE - текущая дата
WHERE date > SYSDATE - INTERVAL '8' YEAR >>> от сегодня за 8 лет

Агрегирующие функции
COUNT(), MAX(), MIN(), SUM(), AVG(), LISTAGG()

Порядок выполнения
1. FROM
2. WHERE
3. SELECT
4. GROUP BY
5. HAVING
6. ORDER BY
7. LIMIT

select * from tableName;
select columeName1, columeName2 from tableName;

WHERE MSRPRD_DATE >= TO_DATE('01.11.2019', 'DD.MM.YYYY')

select COUNT(*) from tableName;
select SUM(columeName) from tableName;
select AVG(columeName) from tableName;
select MIN(columeName) from tableName;
select MAX(columeName) from tableName;

select columeName1, columeName2 from tableName ORDER BY columeName1 asc/desc; //сортировка по возрастанию/убыванию

//неповторяющиеся данные
select DISTINCT columeName1 from tableName;
select COUNT(columeName1), columeName2 from tableName GROUP BY columeName1;

//фильтр
select * from tableName WHERE **ColumeName1** {= > < !=} **value1 {**and or} **ColumeName2** {= > < !=} **value2**;
select * from tableName WHERE **ColumeName** between **value1** and **value2**;
select * from tableName WHERE **ColumeName** like {'%ex%', '_ext'}; //not like
select * from tableName WHERE **ColumeName** in (value1, value2…);
select * from tableName WHERE **ColumeName** is null; //is not null

select COUNT(columeName1), columeName2 from tableName GROUP BY columeName1 HAVING COUNT(columeName1) > 1;
GROUP BY TRUNC(date)
//вложенный запросп
select * from tableName WHERE **ColumeName** between **value1** and **value2** IN (
select * from tableName WHERE **ColumeName** between **value1** and **value2**

);

левое соединение - все строки из T1, даже если их нет в T2
правое соединение - все строки из T2, даже если их нет в T1
внутреннее соединение - строки которые есть и в T1 и в T2

```
select columeName1, columeName2 from tableName1
JOIN tableName2 ON t1.columeName = t2.columeName;

select columeName1, columeName2 from tableName1
LEFT OUTER JOIN tableName2 ON t1.columeName = t2.columeName;

select columeName1, columeName2 from tableName1
RIGHT OUTER JOIN tableName2 ON t1.columeName = t2.columeName;
```

теория http://moodle.it-academy.by/course/index.php?categoryid=1
практика http://www.sql-ex.ru/learn_exercises.php
безопасность
https://ru.wikipedia.org/wiki/%D0%92%D0%BD%D0%B5%D0%B4%D1%80%D0%B5%D0%BD%D0%B8%D0%B5_SQL-%D0%BA%D0%BE%D0%B4%D0%B0

```
//show 1 row
SELECT *
FROM (
   SELECT *
   FROM EDATA_PORTAL.VDOCUMENT_ALL
   WHERE MSRPRD_DATE = '15.09.2020'
   AND CONTRACT_ID IS NOT NULL
   AND CONTRACT_NUMBER IS NOT NULL
   AND KEKV IS NOT NULL
   AND KPK IS NOT NULL
   AND BUDGET_CODE IS NOT NULL
   )
WHERE ROWNUM = 1
;
```

SELECT

SELECT * FROM table_name;
SELECT colum(s)_name FROM table_name;

select **DISTINCT** colum(s)_name from table_name;

select * from table_name WHERE condition(s);
select * from table_name where age > >= < <= = <> != 5;
select * from table_name where age **BETWEEN** 5 AND 50;
select * from table_name where E age **IN** (5, 10);
select * from table_name where E age **IS NULL** / IS NOT NULL;
select * from table_name where name **LIKE** 'A%';
select * from table_name where name LIKE 'ap_a';
select * from table_name where name LIKE 'Dr_%' escape '\'; (Dr_Anna)

select * from table_name where age <18 **AND** name LIKE 'A%';
select * from table_name where age <18 **OR** age>65;
select * from table_name where name **NOT LIKE** 'A%';
select * from table_name where NOT name = 'Anna';

select * from table_name **ORDER BY** name;
select * from table_name order by name **DESC**;
select * from table_name order by name **NULLS FIRST** / NULLS LAST;

Character functions

LOWER(string) = to small symbol
UPPER(string) = to big symbol
INITCAP(string) = to big first symbol in the each word

CONCAT(string, string) = to concatanation two words
SELECT colum_name1 ||' '|| colum_name2 FROM table_name; = to concatanation

LENGTH(field) = get length from the word

LPAD(s,n,p) = add to left symbol p, where n – length new word
RPAD(s,n,p) = add to right symbol p, where n – length new word
select LPAD('123',4, '0') from dual; = 123 / 0123

TRIM()
select trim(' Zaur ') from dual; = delete all space
select trim(trailing 'q' from 'Zaurqqqq') from dual; = delete all end ‘q’
select trim(leading 'q' from 'qqZaurqqqq') from dual; = delete all begin ‘q’
select trim(both 'q' from 'qqZaurqqqq') from dual; = delete all begin/end ‘q’

INSTR()
select instr('Zaur', 'u') from dual; = get first position for ‘u’
select instr('Zaur Tregulov', 'u ',4) from dual; = where 4 – start position search
select instr('Zaur Tregulov', 'u',1,2) from dual; = where 2 – second position for ‘u’

SUBSTR(string, start position, number of characters)
select substr('Zaur',2) from dual; = aur
select substr('Zaur',2,2) from dual; = au

REPLACE(string, search item, replacement item)
select replace('Zaur, privet', 'privet') from dual; = Zaur,
select replace('Zaur, privet', 'privet', 'poka') from dual; = Zaur, poka
select replace('Zaur, privet', 'i', '*') from dual; = Zaur, pr*vet

Numeric functions

ROUND(n, presition)
select round(3.14) from dual; = 3
select round(3.16, 1) from dual; = 3.2
select round(3568, -1) from dual; = 3570

TRUNC(n, presition)
select trunc(3.14) from dual; = 3
select trunc(3.16, 1) from dual; = 3.1
select trunc(3568, -1) from dual; = 3560

MOD(dividend, divisor)
select mod(5, 2) from dual; = 1 остаток от деления

Date functions

SYSDATE
select sysdate from dual; = get today date

MONTHS_BETWEEN(end_date, start_date)
ADD_MONTHS(date, number_of_months)

NEXT_DAY(date, day_of_the_week)
LAST_DAY(date)

ROUND(date, date precision format)
TRUNC(date, date precision format)
date precision format = centorial-CC, year-YYYY, quarter-Q, month-MM, week-W, day-DD, hour-HH, minut-MI

Conversion functions: to_char(), to_date(), to_number(), nvl()

TO_CHAR(**number**, format_mask, nls_parameters)
select '$'||to_char(15) from dual; = $15
format_mask = '99999' - 18
format_mask = '099999' - 000018
format_mask = '099999.999' - 000018.350
format_mask = '099999D999' - 000018.350
format_mask = '099,999,999' - 001,234,567
format_mask = '099999G999' – 001,234,567
format_mask = '$099999' - $000018 (dollar change)
format_mask = 'L099999' - $000018 (local change)
format_mask = '099999MI' - 000018- (end minus for negative number)
format_mask = '099999PR' - <000018> (<> for negative number)
format_mask = 'S099999' - +000018 (show sign of number)

TO_CHAR(**date**, format_mask, nls_parameters)
select to_char('20.10.2019') from dual;
select to_char(sysdate,'Month','NLS_DATE_LANGUAGE = AMERICAN') from dual;
format_mask = 'Y' – 9
format_mask = 'YY' - 19
format_mask = 'YYY' - 019
format_mask = 'YYYY' - 2019
format_mask = 'RR' - 19
format_mask = 'YEAR' - twenty nineteen
format_mask = 'MM' - 10
format_mask = 'MON' - OCT
format_mask = 'MONTH' - OCTOBER
format_mask = 'D' – 6 day number
format_mask = 'DD' – 20 day of month
format_mask = 'DDD' – 263 day of year
format_mask = 'DY' – FRI day of week
format_mask = 'DAY' – FRIDAY full day of week
format_mask = 'W' – 3 week of month
format_mask = 'WW' – 38 week of year
format_mask = 'Q' – 3 quarter of year
format_mask = 'CC' – 21 centure
format_mask = 'PM' – AM/PM show for time
format_mask = 'HH24' – show time in format 24 hour

TO_DATE(**text**, format_mask, nls_parameters)
select to_date('08-10-19', 'dd-mm-YYYY') from dual;
select to_char(to_date('08-10-19', 'yy-mm-dd'),'dd-MON-yyyy hh24:mi:ss') from dual;

TO_NUMBER(**text**, format_mask, nls_parameters)
select to_number('4555,77') from dual;
select to_number('4555.77', '9999.99') from dual;
select to_number('$4555.77', '$9999.99') from dual;
select to_number('4,555.77', '9,999.99') from dual;
select to_number('<4,555.77>', '9,999.99PR') from dual;

NVL(value, ifnull)
select nvl(18, 19) from dual;
select nvl(null, 19) from dual;

NVL2(value, ifnotnull, ifnull)
select nvl2(18, 19, 20) from dual;
select nvl2(null, 19, 20) from dual;

NULLIF(value1, value2)
select nullif(18, 19) from dual;
select nullif(19, 19) from dual;

COALESCE(value1, value2, …, valueN)
select coalesce(1, null, 2) from dual;
select coalesce(null, null, 2) from dual;

Conditional functions: if – then – else

DECODE(**expr, comp1, iftrue1**, comp2, iftrue2, …, compN, iftrueN, iffalse)
select decode(3*4, 12, 'True', 'False') from dual;

//simple (swith)
CASE expr
WHEN comp1 THEN iftrue1
WHEN comp2 THEN iftrue2
…
WHEN compN THEN iftrueN
ELSE iffalse
END

```
select
   case 3*4
   when 12 then 100
   end
from dual;
```

//searched
CASE
WHEN cond1 THEN iftrue1
WHEN cond2 THEN iftrue2
…
WHEN condN THEN iftrueN
ELSE iffalse
END

```
select
   case
   when 3*4=12 then 100
   end
from dual;
```

Group functions: count(), sum(), avg(), max(), min(), group by

COUNT(1) – only number
select count(1) from table_name;
select count(field_name) from table_name; = **null not count**
select count(field_name) from table_name where field_name < 5;
select count(distinct field_name) from table_name; = **uniq value count**

SUM(field_name) – only number
select sum(field_name) from table_name; = **null not sum**
select sum(distinct field_name) from table_name; = **uniq value sum**

AVG(field_name) – only number
select avg(field_name) from table_name; = **null not avg**
select avg(distinct field_name) from table_name; = **uniq value avg**

MAX(field_name) –number and string
select max(field_name) from table_name;

MIN(field_name) – number and string
select min(field_name) from table_name;

GROUP BY
select department_id, count(1) from employees group by department_id;
select department_id, count(1) from employees group by department_id HAVING count(1) > 5;

SUBQUERY

```
select first_name, salary
from employees
where salary > (select avg(salary) from employees)
;

select
        (select min(salary) from jobs) min_zp,
        (select max(length(first_name)) from employees) max_length_name
from dual
;
```

You need the single result subquery if using = < <= > >= <>.
For multi result subquery you need **IN, NOT IN, >ANY, >ALL**.

```
        //correlation subquery
select t1.fname from employee t1
where t1.salary >
        (select avg(t2.salary)
        from employee t2
        where t2.department_id = t1.department_id)
;
```

******* ******* *******

INNER JOIN: join on, self join

inner join **has not null fields**

NATURAL JOIN (EQUIJOIN) – tables have the **total field name**
select * from table1
natural join table2;

JOIN USING (EQUIJOIN)
select column(s) from table1
join table2 using (column(s));

JOIN ON (EQUIJOIN)
select column(s) from table1 as t1
join table2 as t2 on t1.field = t2.field;

select column(s) from table1 as t1
join table2 as t2 on (t1.fieldA=t2.fieldA and t1.fieldB=t2.fieldB)
where salary>500;

SELF JOIN – for structure info
select column(s) from table1 as t1
join table1 as t2 on t1.field = t2.field;

NONEQUIJOIN
select column(s) from table1 as t1
join table1 as t2 on (t1.field = t2.field and salary*2<max_salary); //< <= > >=

OUTER JOIN: left, right, full, cross

outer join **has null fields**

LEFT OUTER JOIN
select column(s) from table1 as t1
left outer join table2 as t2 on t1.field = t2.field;

RIGHT OUTER JOIN
select column(s) from table1 as t1
right outer join table2 as t2 on t1.field = t2.field;

FULL OUTER JOIN
select column(s) from table1 as t1
full outer join table2 as t2 on t1.field = t2.field;

CROSS JOIN
select column(s) from table1
cross join table2;

Вертикальное объединение
- объединение
- вычитание

Горизонтальное объединение
- INNER JOIN = X,Y,K + Z,X,M = X
- LEFT JOIN = X,Y,K + Z,X,M = X,Y.K
- RIGHT JOIN = X,Y,K + Z,X,M = Z,X,M
- FULL JOIN = X,Y,K + Z,X,M = X,Y,K,Z,M
- CROSS JOIN = X,Y,K + Z,X,M = XZ, XX, XM, YZ, YX, YM, KZ, KX, KM

UNION, INTERSECT, MINUS

UNION ALL – 5 + 4 = **9**

```
select * from t1 where salary > 5
union all
select * from t2 where salary < 25
;
```

UNION – (sort + set) 5 + 4 = **7**

```
select salary from t1
union
select salary from t2
order by salary desc
;
```

INTERSECT – (sort + set) 5 + 4 = **2**

```
select salary from t1
intersect
select salary from t2
;
```

EXCEPT / MINUS – (sort + set) 5 – 4 = **3**; 4 – 5 = **2**

```
select salary from t1
minus
select salary from t2
;
select salary from t2
minus
select salary from t1
;
```

Count t1.fields must be equals t2.fields.

******* ******* *******

DML: insert, update, delete, merge

INSERT
insert into table_name (columns list)
values(value);

insert into countries (country_id, country_name, region_id)
values('SW', 'Sweden', 1);

UPDATE
update table_name
set column(s) = value(s) where condition(s);

update employees
set salary=1000 where employee_id=15;

DELETE
delete from table_name
where condition(s);

delete from new_emps
where age=24;

MERGE
merge into new_emps ne
using employees e
on (ne.emp_id=e.employee_id)
when matched then
update set ne.start_date=sysdate
delete where ne.job like '%IT%'
when not matched then
insert (emp_id, name, start_date, job)
values (employee_id, last_name, hire_date, job_id);

TCL: commit, rollback, savepoint

transaction = DML commands list.

COMMIT – show update for all sessions (users)
transaction;
commit;

ROLLBACK
transaction;
rollback;

transaction;
rollback to savepoint savepoint_name;

SAVEPOINT
savepoint savepoint_name;

AUTOCOMMIT – **do not use**
set autocommit on;
set autocommit off;

LOCK
select * from employee for update;

DDL: create, alter, rename, truncate, drop

CREATE

```
create table students(
student_id integer,
name varchar2(15)
);
```

ALTER

```
alter table table_name
add (column_name data_type default expr);

alter table table_name
modify (column_name data_type default expr);

alter table table_name read only;

alter table table_name
drop column column_name;

alter table table_name
set unused column column_name;

alter table table_name
drop unused columns;
```

RENAME

```
alter table table_name
rename column column_name1 to column_name2;
```

TRUNCATE

```
truncate table table_name;
```

DROP

```
drop table shema.table_name;
```

******* ******* *******

Constraint, Sequence, Index

constraint – business rools: unique, not null, primary key, foreign key, check

UNIQUE CONSTRAINT
принуждает столбец(цы) содержать только уникальные значения (отсутствие дубликатов в столбце). Исключение – null.

NOT NULL CONSTRAINT
не разрешает столбцам содержать значение null.

PRIMARY KEY CONSTRAINT
принуждает столбец(цы) содержать только уникальные значения (отсутствие дубликатов в столбце) и не разрешает содержать значение null.

FOREIGN KEY CONSTRAINT
принуждает использовать только значения из определенного столбца таблицы-родителя или значение null. Связывает две таблицы между собой.

CHECK CONSTRAINT
принуждает использовать только те значения, которые удовлетворяют указанному условию(ям).

SEQUENCE
генерирование уникальных значений по заранее определенному расчету. Используется для primary key.

INDEX – set automation for primary key, unique

B-TREE INDEX – default, используется:
- когда много строк;
- когда количество строк в оутпуте составляет 2%-4% от общего кол-ва строк;
- когда много уникальных значений;
- когда используется WHERE, JOIN по столбцу.

BITMAP INDEX используется:
- когда много строк;
- когда мало уникальных значений;
- когда используется AND, OR, NOT по столбцу.

View, Synonym

VIEW – select to table
-безопасность / ограничение (доступ к 3 полям из 5);
-упрощение написания запроса (не нужен JOIN);
-предотвращение ошибок;
-понятные названия полей;
-перфоменс.

SIMPLE VIEW
-one table
-no functions
-no aggregation

COMPLEX VIEW
-join tables
-functions
-aggregation

SYNONYM
предоставляет таблице другое имя.

******* ******* *******

ПОДСТАНОВКИ

select name, salary from employees where employee_id = 30; select name, salary from employees where name = 'Steven';	select name, salary from employees where employee_id = &ID; select name, salary from employees where name = '&name';