

Міністерство освіти і науки України
Запорізький національний технічний університет

**ПРОГРАМА, МЕТОДИЧНІ ВКАЗІВКИ ТА
КОНТРОЛЬНІ ЗАВДАННЯ
з дисципліни
«АДМІНІСТРУВАННЯ КОМП'ЮТЕРНИХ
СИСТЕМ»**

для студентів спеціальності 121 – «Інженерія програмного
забезпечення» заочної форми навчання



2017

Програма, методичні вказівки та контрольні завдання з дисципліни «Адміністрування комп'ютерних систем» для студентів спеціальності 121 – «Інженерія програмного забезпечення» заочної форми навчання /Уклад.: С.К. Корнієнко.– Запоріжжя: ЗНТУ, 2017. – 13 с.

Укладач: С.К. Корнієнко, доцент, к.т.н.

Рецензент: О.О.Степаненко, доцент, к.т.н.

Відповідальний
за випуск: С.О.Субботін, професор, д.т.н.

Затверджено
на засіданні кафедри
"Програмні засоби"

Протокол № 1 від 28.08.2017 р.

1. МЕТА ТА ЗАДАЧІ ДИСЦИПЛІНИ, ЇЇ МІСЦЕ В НАВЧАЛЬНОМУ ПРОЦЕСІ

1.1. Мета викладання дисципліни

Метою вивчення дисципліни є теоретична й практична підготовка студентів в області адміністрування комп'ютерних систем у такому ступені, щоб вони могли вибирати необхідні технічні, алгоритмічні, програмні та технологічні рішення, вміти пояснити принципи їх функціонування та правильно їх використовувати.

1.2. Задачі вивчення дисципліни

В наслідку вивчення дисципліни студенти повинні:

знати:

- принципи побудови систем адміністрування та управління;
- програмну структуру, протоколи й служби, інформаційні бази даних систем адміністрування;
- сучасні методи та засоби розробки таких систем;

вміти:

- використовувати методи моделювання при виборі структури систем адміністрування;
- використовувати методи та засоби інформаційних і телекомунікаційних технологій.

1.3. Зв'язок з іншими дисциплінами

Вивчення даної дисципліни базується на знаннях, одержаних при вивченні дисциплін «Конструювання програмного забезпечення», «Бази даних» і «Людиномашинна взаємодія». Отримані знання можуть використовуватися у подальшому навчанні, дипломному проектуванні, а також професійній виробничій та науковій діяльності випускника.

1.4. Загальні вказівки

Дана дисципліна вивчається у сьомому семестрі й містить 6 лекційних годин і 6 годин лабораторних робіт, а також виконання контрольної роботи.

Працювати над дисципліною треба систематично. Під час самостійної опрацювання матеріалу слід також враховувати, що контрольні завдання складені таким чином, щоб результат їхнього виконання був покладений у підвалини виконання лабораторного практикуму.

Підведення підсумків вивчення дисципліни здійснюється у вигляді заліку. До складення заліку допускаються студенти, які успішно виконали лабораторний практикум і контрольну роботу..

2. ПРОГРАМА ДИСЦИПЛІНИ ТА МЕТОДИЧНІ ВКАЗІВКИ З ЇЇ ВИВЧЕННЯ

Тема 1. Поняття комп'ютерної системи.

- 1.1. Поняття комп'ютерної інформаційної системи
- 1.2. Принципи побудови автоматизованих систем
- 1.3. Класифікація автоматизованих систем

Тема 2. Програмна структура систем адміністративного керування.

- 2.1 Поняття інфраструктури інформаційної системи
- 2.2 Задачі та функції адміністрування в інформаційних системах
- 2.3 Системне адміністрування

Тема 3. Функції й функціональні області адміністративного керування.

- 3.1 Мережне адміністрування
- 3.2 Методи й завдання адміністратора мережі
- 3.3 Автоматизація керування мережею
- 3.4 Архітектура Клієнт-сервер
- 3.5 Адміністрування в корпоративних мережах

Тема 4. Керування безпекою.

- 4.1 Безпека мережі
- 4.2 Призначення й функціонування брандмауера
- 4.3 Призначення проксі-сервера

Під час опрацювання матеріалу необхідно користуватися не тільки основною, а й додатковою літературою.

3. КОНТРОЛЬНА РОБОТА

3.1 Загальні положення

В процесі вивчення дисципліни студенти заочної форми навчання повинні виконати контрольну роботу. Основна ціль її виконання – закріплення теоретичного матеріалу та оволодіння навичками його практичного використання.

Контрольна робота передбачає виконання двох завдань, перше з яких присвячене роботі з привілеями, а друге роботі з поданнями.

Приступати до виконання контрольних робіт слід лише після детального вивчення теоретичного матеріалу та виконання лабораторних робіт згідно з відповідними методичними вказівками.

Результати виконання кожної контрольної роботи оформлюються у вигляді звіту, який виконується комп'ютерним способом і повинен містити завдання та тексти відповідних SQL-запитів.

Робота повинна бути датована та підписана виконавцем.

Виконані контрольні роботи повинні бути представлені на перевірку в суворій відповідності до графіку навчального процесу. При наявності зауважень рецензента слід виконати всі виправлення та доповнення так, щоб при повторному рецензуванні можна було б співвіднести попереднє та нове рішення.

Захист контрольних робіт відбувається безпосередньо на комп'ютері.

3.2 Основні теоретичні відомості

3.2.1 Привілеї

У багатокористувацькій системі питання безпеки даних вирішуються на рівні SQL і СКБД. Для цього є два механізми.

Перший заснований на *виділенні повноважень*. При цьому за допомогою команд **GRANT** і **REVOKE** указується, яким користувачам дається право виконувати визначені команди у відношенні визначених таблиць, курсорів і стовпчиків.

Другий механізм заснований на *використанні подань* у сполученні з командами **GRANT** і **REVOKE** для надання виборчого доступу до різноманітних підмножин даних.

Обидва механізми припускають, що в системі передбачений засіб

ідентифікації (розпізнавання) кожного конкретного користувача. Коли користувач створює таблицю або подання, він автоматично стає їхнім володарем і отримує по відношенню до них повний набір привілеїв. Для надання доступу до цих об'єктів іншим користувачам необхідно, щоб володар явним чином надав їм ці права.

Одиницею даних або "об'єктом даних", для яких повинна бути створена система безпеки, може бути як уся база цілком або який-небудь набір відношень, так і деяке значення даних для заданого атрибута у визначеному відношенні.

Зазвичай в момент входу до системи користувач повинен ввести не тільки свій **ідентифікатор** (наприклад, ім'я або посада), але також і **пароль**.

У системі можуть підтримуватися **групи користувачів** і забезпечуватися однакові права доступу для користувачів однієї групи, наприклад, для всіх осіб із розрахункового відділу. Крім того, операції включення нових користувачів до групи або їхнє вилучення з неї можуть виконуватися незалежно від операції завдання привілеїв для цієї групи.

🔍 **Привілеї (privileges)**— це сукупність дій, які дозволено виконувати користувачеві по відношенню до конкретної таблиці або подання.

В стандарті ISO визначається такий набір привілеїв:

- **SELECT** — право вибирати дані з таблиці;
- **INSERT** — право додавати до таблиці нові рядки;
- **UPDATE** — право змінювати дані в таблиці;
- **DELETE** — право вилучати рядки з таблиці;

Команда **GRANT** призначає привілею і має два варіанти синтаксису, що відрізняються місцем розташування списку стовпчиків:

- 1) **GRANT {ALL | список_повноважень }
ON {ім'я_таблиці [(список_стовпчиків)] |
ім'я_курсуру [(список_стовпчиків)]}
TO { PUBLIC | список_користувачів}
[WITH GRANT OPTION]**
- 2) **GRANT {ALL | список_повноважень [(список_стовпчиків)]}
ON {ім'я_таблиці | ім'я_курсуру }
TO { PUBLIC | список_користувачів}
[WITH GRANT OPTION]**

Команда **REVOKE** анулює видані повноваження:

```

REVOKE {ALL | список_повноважень }
      ON {ім'я таблиці [(список_стовпчиків)] |
        ім'я курсору [(список_стовпчиків)]}
FROM { PUBLIC | список_користувачів }
[WITH GRANT OPTION]

```

```

REVOKE {ALL | список_повноважень [(список_стовпчиків)]}
      ON {ім'я таблиці | ім'я курсору }
FROM { PUBLIC | список_користувачів }
[WITH GRANT OPTION]

```

У списку повноважень вказується або ключове слово **ALL**, або список повноважень, якими наділяється користувач. Повноваження в списку розділяються комами.

У операторі **ON** вказується таблиця або курсор, на які видаються або анулюються повноваження (по одній таблиці або одному курсору на оператор). Коли повноваження видаються на окремі стовпчики, їхні імена вказуються в круглих дужках. При цьому стовпчики повинні належати одній таблиці або курсору. Якщо список стовпчиків не зазначений, повноваження видаються або анулюються на всю таблицю або курсор.

Ключове слово **PUBLIC** ("загальнодоступний") означає всіх користувачів у системі. Альтернативою йому є список користувачів.

Наприклад, **Іванов** одержує дозвіл додавати дані до таблиці **товар** і оновлювати її:

```

grant insert, update
on товар
to Іванов;

```

Для заборони Іванову оновлювати стовпчики **дисконт** і **ціна** у таблиці **товар**:

```

revoke update (дисконт, ціна)
on товар
from Іванов;

```

```

revoke update
on товар (дисконт, ціна)
from Іванов;

```

Необов'язковий оператор **WITH GRANT OPTION** визначає, чи може користувач, який наділяється повноваженнями, у свою чергу передавати їх іншим користувачам. Наприклад наступний оператор наділяє користувача **Петрова** правом виконувати операцію **SELECT** у таблиці **Замовлення** і дозволяє йому передавати це право іншим користувачам:

*grant select
on Замовлення
to Петров
with grant option;*

3.2.2 Використання подань при адмініструванні баз даних

Бувають ситуації, коли користувачеві необхідно дозволити доступ не до всієї інформації, що зберігається в таблиці, а лише до її частини: наприклад, необхідно, щоб продавці могли переглядати таблицю Salespeople, але не могли бачити комісійні своїх колег. Або ситуація, коли щодня необхідно одержувати інформацію на основі різних таблиць за допомогою складних запитів. Для того, щоб не повторювати щораз складний запит, можна створити на його основі подання (VIEW).

Подання – це динамічний результат однієї або декількох реляційних операцій над базовими відношеннями з метою створення деякого іншого відношення.

Подання є **віртуальним відношенням (virtul relation)**, якого реально в базі даних не існує, але яке створюється на вимогу окремого користувача в момент надходження цієї вимоги.

З погляду користувача подання є відношенням, що постійно існує та з яким можна працювати, як із базовим відношенням.

Однак подання не зберігається в базі даних так, як базові відношення (хоча його визначення зберігається в системному каталозі). Вміст подання визначається як результат виконання запиту до одного або декількох базових відношень.

Подання – це вікно, через яке користувач може побачити інформацію з БД. При зверненні до подання виконується запит до базової таблиці або таблиць, на основі яких створено подання.

Наприклад, необхідно дозволити покупцям бачити в таблиці **Товари** тільки дані про назву товару та ціну відпускну. У цьому випадку можна створити подання **Прайс-лист** на основі таблиці **Товари**, а потім дозволити всім покупцям читати інформацію з цього подання.

*create view Прайс-лист
as
select назва, ціна_відпускна as ціна*

from Товари;

Зверніть увагу на те, що в пропозиції CREATE VIEW після імені подання не зазначені імена стовпців, з яких воно буде складатися. Тому беруться імена стовпців базової таблиці або їхні псевдоніми, на основі яких будується подання. Якщо при створенні подання використовуються агрегатні функції, то такі стовпці повинні бути названі в пропозиції CREATE VIEW.

Подання, засновані на запиті, що використовує *речення GROUP BY*, називаються *груповими*. При цьому допускається використання речення *HAVING*.

При створенні подання можна використовувати з'єднання таблиць. Наприклад, в БД є три таблиці:

Покупці(Код,Прізвище,Тел),
Продавці (Код, Прізвище),
Замовлення (Ном, Покупець, Дата, Вартість, Продавець).

Необхідно визначити за минулий місяць всіх покупців з їхніми замовленнями та продавців, які їх обслуговували:

```
create view Покупці_Замовлення
as
select П.Прізвище as Покупець, Пр.Прізвище as Продавець,
З.Дата, З.Вартість
from Замовлення З, Продавці Пр, Покупці П
where З.Покупець=П.Код and З.Продавець=Пр.Код
and month(date()-month (З.Дата)=1;
```

Це подання побудовано на основі запиту, що з'єднує три таблиці. Звернутися до нього можна також через запит. Наприклад, нам необхідно одержати всі замовлення покупця Іванова в минулому місяці:

```
select *
from Покупці_Замовлення
where Покупець='Іваненко';
```

Будь-які операції над поданням автоматично транслюються в операції над відношеннями, на підставі яких це подання було створено. Подання мають **динамічну** природу, тобто зміни в базових відносінах, які можуть вплинути на вміст подання, негайно відбиваються на вмісті цього подання.

Якщо користувачі вносять у подання деякі припустимі зміни, ці зміни негайно заносяться до відповідних базових таблиць.

На підставі таблиці **Продавці** (Код, Прізвище, Місто, Рейтинг) створимо подання:

```
create view Ton_Продавці
as
select *
from Продавці
where Рейтинг=300;
```

А тепер виконаємо команду оновлення

```
insert into Ton_Продавці
values (201, 'Гриценко', 'Запоріжжя', 250);
```

Ця команда виконається й у базову таблицю буде вставлений запис, але до подання запис не буде доданий, тому що значення Рейтинг не рівно 300.

Для того, щоб відслідковувати такі ситуації, можна скористатися реченням **WITH CHECK OPTION**, яке вставляється в команду створення подання.

```
create view Ton_Продавці
as
select *
from Продавці
where Рейтинг=300;
with check option;
```

Тепер така команда **INSERT** буде відхилена.

3.3 Зміст контрольної роботи

Зміст конкретного завдання визначається викладачем індивідуально для кожного студента.

ЛІТЕРАТУРА

Базова

1. Корнієнко С.К. Проектування інформаційного забезпечення автоматизованих систем. Навч. Посібник/ С.К.Корнієнко. – Запоріжжя: ЗНТУ, 2015. – 232 с.
2. Кенин А.М.Практическое руководство системного администратора / А.М.Кенин. – СПб: БХВ-Петербург, 2013. – 523 с
3. Васвани Д. MySQL: Использование и администрирование / Д. Васвани. – СПб.: Питер, 2011. – 368 с.
4. Буров Є.В Комп'ютерні мережі / Є.В.Буров. – К.: Ліра-К, 2012. – 262с
5. Романовський Ю.Р. Адміністрування комп'ютерних мереж і систем: Навч. пос. / Ю.Р. Романовський, В.В.Олексюк, А.В. Балик. – Тернопіль: Навчальна книга – Богдан, 2010. – 196 с.

Допоміжна

6. Блэк Ю. Проектирование защищенных информационных технологий /Ю. Блек. – СПб: Вильямс, 2012. – 258 с.
7. Левченко О.М. Основи Інтернету / О.М. Левченко. – К.: Видавнича група ВНУ, 2009, – 285 с.

